DOCUMENT RESUME

ED 233 686 IR 010 776

AUTHOR	Goldfine, Alan H., Ed.
TITLE	Data Base Directions: Information Resource Management - Strategies and Tools. Proceedings of the Workshop of the National Bureau of Standards and the Association for Computing Machinery (Ft. Lauderdale, Florida, October 20-22, 1980).
INSTITUTION	Association for Computing Machinery, New York, N.Y.; National Bureau of Standards (DOC), Washington, D.C. Inst. for Computer Sciences and Technology.
REPORT NO	NBS-SP-500-92
PUB DATE	Sep 82
NOTE	186p.
AVAILABLE FROM	Superintendent of Documents, U.S. Government Printing Office, Washington, DC 20402 (1982-360-997/2238, $6.50).
PUB TYPE	Collected Works - Conference Proceedings (021) -- Viewpoints (120)
EDRS PRICE	MF01/PC08 Plus Postage.
DESCRIPTORS	*Databases; *Data Processing; Information Needs; *Information Services; *Management Information Systems; Needs Assessment; Organizational Communication; Policy Formation; Program Development
IDENTIFIERS	Database Design; *Data Dictionary Systems; Data Management; *Information Resource Management

ABSTRACT

This workshop investigated how managers can evaluate, select, and effectively use information resource management (IRM) tools, especially data dictionary systems (DDS). An executive summary, which provides a definition of IRM as developed by workshop participants, precedes the keynote address, "Data: The Raw Material of a Paper Factory," by John A. Gosden. The reports from four working panels which follow cover: (1) the uses of an information resource dictionary system (IRDS), with a discussion of organizational information systems and database perspectives, as well as tables illustrating how the IRDS is used in various system life cycle (SLC) phases and the relationships between IRDS functions and its users/uses; (2) IRM policies and controls, with a sample strategic system architecture plan and discussions of IRM policies applicable across all SLC phases and other policies for specific phases; (3) logical database design, with discussions of user requirements analysis, information modeling, the interface between logical and physical design, and the role of DDS in the management of data, as well as an 84-item bibliography; and (4) physical database design, with a discussion of six SLC phases, including the user requirements analysis, physical design, implementation, operations, evolution, and end game phases. A list of workshop participants and their organizational affiliations concludes the report. (ESR)

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

- This document has been reproduced as received from the person or organization originating it.
- ☐ Minor changes have been made to improve reproduction quality.

- Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

ED233686

Computer Science and Technology

NBS Special Publication 500-92

Data Base Directions
Information Resource Management — Strategies and Tools

Proceedings of the Workshop of the National Bureau of Standards and the Association for Computing Machinery, held Oct. 20-22, 1980, at Fort Lauderdale, FL.

Alan H. Goldfine, Editor

Center for Programming Science and Technology
Institute for Computer Sciences and Technology
National Bureau of Standards
Washington, DC 20234

Major Contributors: Mayford L. Roark, General Chairperson; Henry C.Lefkovits, John K. Lyon, Shamkant B. Navathe and Anthony J. Winkler

Association for Computing Machinery
acm

U.S. DEPARTMENT OF COMMERCE
Malcolm Baldrige, Secretary

National Bureau of Standards
Ernest Ambler, Director

Issued September 1982

IR010776

6.50 ııı 2

Reports on Computer Science and Technology

The National Bureau of Standards has a special responsibility within the Federal Government for computer science and technology activities. The programs of the NBS Institute for Computer Sciences and Technology are designed to provide ADP standards, guidelines, and technical advisory services to improve the effectiveness of computer utilization in the Federal sector, and to perform appropriate research and development efforts as foundation for such activities and programs. This publication series will report these NBS efforts to the Federal computer community as well as to interested specialists in the academic and private sectors. Those wishing to receive notices of publications in this series should complete and return the form at the end of this publication.

Library of Congress Catalog Card Number: 82-600584

National Bureau of Standards Special Publication 500-92
Natl. Bur. Stand. (U.S.), Spec. Publ. 500-92, 174 pages (Sept. 1982)
CODEN: XNBSAV

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1982

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402
Price $6.50
(Add 25 percent for other than U.S. mailing)

PREFACE

This report constitutes the proceedings of a three-day workshop on information resource management tools, held in Fort Lauderdale, Florida on October 20-22, 1980. The workshop was sponsored jointly by the Institute for Computer Sciences and Technology of the National Bureau of Standards (NBS) and the Association for Computing Machinery (ACM). The workshop continues the close working relationship that was started in 1972 between the Institute and the ACM.

The first workshop in this series was Data Base Directions: The Next Steps, held in October, 1975. NBS published the workshop report as Special Publication 451, and it was reprinted both by the ACM Special Interest Group on the Management of Data and the ACM Special Interest Group on Business Data Processing. The British Computer Society published the report for European distribution, and it was excerpted by the IEEE and Auerbach.

The second workshop, Data Base Directions: The Conversion Problem, was held on November 1-3, 1977. It addressed the questions: "What information can help a manager assess the impact a conversion will have on a database system?" and "What aid will a database system be during a conversion?" The report has been published by NBS as Special Publication 500-64, and as a joint publication of the ACM Special Interest Groups on the Management of Data and Business Data Processing. Portions of the report were presented at the 1978 National Computer Conference.

The purpose of this latest workshop was to generate information that Federal and private industry managers can apply to evaluate and select information resource management tools, especially data dictionary systems, and use them effectively. Such information is becoming increasingly important to Federal agencies working to manage the enormous collections of data characteristic of today's operations.

The workshop divided into four working panels to consider information resource management tools from the standpoint of uses, policies and controls, logical database design, and physical database design. Each panel prepared a draft report, which was then put into final form by the panel chairman for submission to the proceedings editor.

Because the participants in the workshop drew on their personal experiences, they sometimes cited specific vendors and commercial products. The inclusion or omission of a particular company or product does not imply either endorsement or criticism by NBS.

We gratefully acknowledge the assistance of all those who made the workshop possible.

Alan Goldfine, Editor

TABLE OF CONTENTS

Page

PREFACE iii

EXECUTIVE SUMMARY ix

1. INTRODUCTION 3

1.1 THE FIRST TWO DATA BASE DIRECTIONS WORKSHOPS .. 3

1.2 DATA BASE DIRECTIONS III 4
  1.2.1 Using The Data Dictionary System 4
  1.2.2 Standards and Controls 4
  1.2.3 Database Design 4

1.3 CONCLUSION 5

2. DATA: THE RAW MATERIAL OF A PAPER FACTORY 7

2.1 AN INTERESTING CHALLENGE 7

2.2 CHANGES OCCURRING IN TECHNOLOGY 9
  2.2.1 Speech Filing 9
  2.2.2 Electronic Mail and Storage 10
  2.2.3 Image Storage 10
  2.2.4 On-Line Reports, Analysis, Graphics 10
  2.2.5 Retail Databases 10
  2.2.6 Variety and Convenience of Access 11
  2.2.7 Cable TV Extension 11
  2.2.8 One-Stop Processing 11

2.3 GENERAL FORCES STIMULATING CHANGE 12

2.4 THE NEW MANAGER'S WORLD OF 2000 13

3. USES OF THE INFORMATION RESOURCE DICTIONARY SYSTEM FOR IRM 17

3.1 INTRODUCTION 18
  3.1.1 The Objective 18
  3.1.2 A Historical Perspective 19
  3.1.3 The Approach 20

3.2 THE ORGANIZATIONAL PERSPECTIVE 22
  3.2.1 Introduction 22
  3.2.2 The Organization, the IRDS and IRM 22

6

3.2.3 Organizational Users of an IRDS 23
3.2.4 Scope of the Information Resource Entity-types 25

3.3 THE INFORMATION SYSTEMS PERSPECTIVE 26
3.3.1 Introduction 26
3.3.2 Definition of the System Life Cycle 26
3.3.3 Using the IRDS to Support Information Systems Evolution 29
3.3.4 Users of the IRDS 30
3.3.5 Uses of the IRDS 31

3.4 THE DATABASE PERSPECTIVE 33
3.4.1 Approach 33
3.4.2 Users and Uses of the IRDS 34
3.4.3 Enterprise-Oriented Dictionary Function ... 35
3.4.4 Directory Function 36
3.4.5 System Specific Dictionary System 37

3.5 CONCLUSION 38

4. IRM POLICIES AND CONTROLS 49

4.1 INTRODUCTION 50

4.2 DEFINITION OF IRM 50

4.3 SYSTEM LIFE CYCLE 50

4.4 ORGANIZATION OF THE PANEL 51

4.5 IRM POLICY RECOMMENDATIONS 52

4.6 FUNDAMENTAL IRM POLICIES 53

4.7 STRATEGIC SYSTEM ARCHITECTURE AND INFORMATION PLANS 53

4.8 IRM POLICIES APPLICABLE ACROSS ALL SLC PHASES 55
4.8.1 Project Management 56
4.8.2 Configuration Management 56
4.8.3 Metadata Management 57
4.8.4 Documentation Management 58
4.8.5 Review Management 59

4.9 IRM POLICIES FOR SPECIFIC SLC PHASES 59
4.9.1 Requirements Determination Phase 60
4.9.2 Data/System Specification Phase 62
4.9.3 Database/System Design Phase 64
4.9.4 Construction/Testing Phase 66
4.9.5 Integration/System Testing Phase 67

4.9.6 Installation/Operations Phase 68
4.9.7 Maintenance Phase 69
4.9.8 Enhancement and Termination Phases 70

4.10 CONCLUSION AND SUMMARY 70

5. LOGICAL DATABASE DESIGN 73

5.1 INTRODUCTION 74
5.1.1 Organization of the Panel 78

5.2 REQUIREMENTS ANALYSIS 79
5.2.0 Introduction 79
5.2.1 The Activities of the Requirements Phase .. 80
5.2.2 State of the Art 83
5.2.3 Future Directions 88
5.2.4 The Role of the Data Dictionary System in Requirements Analysis 94

5.3 INFORMATION MODELING 99
5.3.1 Introduction 99
5.3.2 The Information Modeling Process 99
5.3.3 Tools for Database Design 103
5.3.4 Database System Communication 113

5.4 INTERFACE OF LOGICAL AND PHYSICAL DATABASE DESIGN 119
5.4.1 Results of Logical Database Design 119
5.4.2 Overlap of Logical and Physical Design ... 120
5.4.3 Alterations of Logical Schemas for Physical Reasons 121

5.5 THE ROLE OF DATA DICTIONARY SYSTEMS IN MANAGEMENT OF DATA 124
5.5.1 Introduction 124
5.5.2 Approach 126
5.5.3 Summary of the Recommendations 126
5.5.4 Dictionary System Implementation 132
5.5.5 Conclusion 133

5.6 REFERENCES 133

6. PHYSICAL DATABASE DESIGN 141

6.1 INTRODUCTION 142

6.2 REQUIREMENTS--METADATA REQUIRED IN DDS 144

6.3 PHYSICAL DATABASE DESIGN PHASE 146
6.3.1 Design Development 147
6.3.2 Physical Allocation 150
6.3.3 Predictive Validation 151

6.4 IMPLEMENTATION 152
  6.4.1 Conversion Plan 152
  6.4.2 Data Dictionary System Assistance 152
  6.4.3 Prototyping 152
  6.4.4 Complexity 153
  6.4.5 Validation of the Design 153
  6.4.6 Data Validation 153
  6.4.7 Stress Testing 153
  6.4.8 Confirmation 154

6.5 OPERATIONS 154
  6.5.1 Gathering Performance Data 154
  6.5.2 Aggregating Performance Data 154
  6.5.3 Reporting Performance Data 155
  6.5.4 Analyzing Performance Data 155
  6.5.5 Performing Maintenance 155
  6.5.6 DDS Usage 155

6.6 EVOLUTION 155
  6.6.1 Definition of Two Types of Reorganization 156

6.7 END GAME 156

7. PARTICIPANTS 159

9

EXECUTIVE SUMMARY

On October 20-22, 1980, the Institute for Computer Sciences and Technology of the National Bureau of Standards (NBS) and the Association for Computing Machinery (ACM) held the third in their series of Data Base Directions workshops. The goal of this workshop was to generate information that managers can apply to evaluate, select, and effectively use information resource management tools.

Among these tools, a central role is played by data dictionary systems (DDS), which have become basic to all phases of data administration. Managers increasingly turn to data dictionary systems to satisfy requirements for information resources management (IRM), to aid in database design, and to provide the information needed for the effective auditing of data.

Therefore, the workshop was organized into four working panels, which met to discuss:

- Uses of the Information Resource Dictionary System for IRM
- IRM Policies and Controls
- Logical Database Design
- Physical Database Design

The workshop participants realized that their first priority was to develop a working definition of information resource management that would allow them to focus on the key issues. The definition that evolved was:

> Information Resource Management (IRM) is whatever policy, action, or procedure concerning information (both automated and non-automated) which management establishes to serve the overall current and future needs of the enterprise. Such policies, etc., would include considerations of availability, timeliness, accuracy, integrity, privacy, security, auditability, ownership, use, and cost-effectiveness.

Uses of the Information Resources Dictionary System for IRM

This panel investigated the question of how data dictionary systems could be used to support managed information environments and help organizations realize the benefits of an enterprise-wide information resource system. After generalizing "data dictionary systems" to "information resource dictionary systems (IRDS)," the panel examined the question from three perspectives:

1. the enterprise's organizational perspective
2. the perspective of information systems supporting the enterprise
3. the perspective of the enterprise's database

From the organizational perspective, the panel concluded that the IRDS should provide information both for those managers directing the business functions of the enterprise, and for those managing its information resources. The IRDS shows functional managers what information is available, the form in which it is available, where and how to obtain it, and its probable cost. Information resource managers need the IRDS for analytical data on how often and how effectively the resources are being used, as well as to help them analyze and assess the impact of proposed changes to an information resource.

The panel then examined the potential uses of the IRDS during the phases of an information system life cycle. For each phase, a table was constructed illustrating how the IRDS is used by various categories of users. For example, during the requirements phase, the information "customer" defines the generic business applications and methodologies, and the information resource manager then translates these to the context of the enterprise's information resources. The overall application is reviewed, through the IRDS, by the data auditor.

The third perspective viewed the IRDS in relation to the development of, access to, and support of enterprise databases--both automated and non-automated. The IRDS is used globally to view all data independently of environment and physical implementation, and to support the functions of conceptual database design, developing prototype systems, and data modeling. Operationally, the IRDS is used as:

- a directory of the enterprise's data and information resources, providing the location (and a path to obtain) the needed resources

- a support for a given database management system (DBMS). Here the IRDS is used by the database administrator to describe and monitor databases, generate schemas, and implement integrity rules and access constraints.

IRM Policies and Controls

This panel was charged with investigating the policies an organization requires to establish standards and controls in an IRM environment. Since the members of the panel had been drawn both from the IRM technology sector and from the internal auditing profession, it was decided to divide the panel into separate technology and auditing working groups. The final report of the panel combines the findings of the two groups.

The panel developed a set of fundamental IRM policies for a hypothetical organization. These policies emphasize the need for an integrated set of planning documents:

- a strategic system architecture plan
- a long range information plan
- guidelines for the use of dictionaries, languages, databases, and networks
- a data standardization plan

After outlining the strategic system architecture plan (the key document) the panel identified specific policies that are important to the overall success of IRM. The report discusses or lists deliverables, applicable computer-based tools, concerns about tools, IRM control policies, and auditing risk exposures. These policies affect all phases of the information system life cycle, and have a significant impact on other areas of an enterprise.

Logical Database Design

This panel addressed the issues of IRM as they relate to the process of logical database design. To best explore this broad area, the panel was divided into four working groups:

1. Requirements Analysis/Assessment of User Needs. This working group closely examined the process by which a meaningful statement of an enterprise's information requirements is developed. The group concluded that existing tools and methodologies in this

area have a number of desirable features, but that no single technique stands out. Data dictionary systems were emphasized as:

- a repository of the information collected during the analysis
- a control mechanism
- a tool to perform basic analyses
- a manager of a database to which more sophisticated analytical tools may be applied

2. Information Modeling. This group identified the basic processes involved in developing an "integrated, formal, implementation-independent specification of application-specific enterprise information." The group surveyed the better known information models and the existing tools for database design. They developed the concept of a "database workshop" to support the design process. This workshop would be built around a data dictionary, which would play an active role as a repository of metadata about the database being designed.

3. Interface between Logical and Physical Design. This group discussed the relationships between logical and physical design. The members reviewed some results of logical design which influence physical design decisions, and examined the impact of physical design decisions on logical database design.

4. Role of Data Dictionaries in Management of Data. This group summarized the role of the DDS as performing the functions of "planning, control, direction, and organization of the information resource." The members concluded that while a data dictionary is integral to a database environment, the DDS itself should be independent of the DBMS. They identified a list of DDS features helpful in designing databases, both in distributed and centralized environments.

Physical Database Design

This panel investigated the physical issues of database design using a data dictionary system. Once again, a system life cycle framework was chosen to describe the uses of the DDS:

1. User Requirements Phase. The dictionary for a new database is initially created during this phase using logical design components such as entities, attributes, and relationships. The collected

"metadata" or data about data is the same as would be collected in a non DDS environment.

2. Physical Design Phase. The panel identified the metadata needed to describe decisions and actions that lead to a specific database design. This metadata is also necessary to determine the distribution and placement of records within the available storage space, and the specification of procedures leading towards validation of the database.

3. Implementation Phase. This phase can be viewed as a conversion from an old system or design to a new one. Certainly the DDS should reflect information about the new design, and possibly, about the older design as well. However, the panel wondered why it shouldn't also be possible to use an active DDS directly to assist in the automatic installation of the database.

4. Operations Phase. The DDS is the vehicle for documenting the answers to these questions:

   * What was the desired performance from the requirements phase?
   * What was the predicted performance from the design phase?
   * What is the actual performance during the operations phase?
   * What are the estimated characteristics of data and usage from the requirements phase?
   * What are the actual characteristics of data and usage during the operations phase?

5. Evolution Phase. Reorganization and redesign may require major changes to the definition of the data. Most of the metadata that the panel identified as necessary during the physical design phase would be needed here also.

6. End Game Phase. The panel recognized the possibility that the operation of a database might at some time be shifted to a distinctly different mechanism. The DDS then would play the role it does during the conversion in the implementation phase. The archiving features of a DDS would be especially important here.

DATA BASE DIRECTIONS
INFORMATION RESOURCE MANAGEMENT--STRATEGIES & TOOLS

Alan H. Goldfine
Editor

This report constitutes the results of a three-day workshop on information resource management tools, held in Fort Lauderdale, Florida on October 20-22, 1980. The workshop was sponsored jointly by the Institute for Computer Sciences and Technology of the National Bureau of Standards (NBS) and the Association for Computing Machinery (ACM).

Patterned after the two previous Data Base Directions workshops, this workshop, Data Base Directions: Information Resource Management -- Strategies & Tools, investigated how managers can evaluate, select, and effectively use information resource management tools, especially data dictionary systems. The approximately seventy workshop participants were organized into four working panels, which met to discuss Uses of the Data Dictionary System, IRM Policies and Controls, Logical Database Design, and Physical Database Design.

Key words: data dictionary system; data management; database; database design; DBMS; information resource management.

1. INTRODUCTION

Mayford L. Roark

GENERAL CHAIRMAN

Biographical Sketch

Mayford L. Roark has headed the information systems function at Ford Motor Company for the past 15 years--first as Assistant Controller, later as Director of Systems, and presently as Executive Director--Systems. Mr. Roark joined the Company's Ford Division in 1952 as a Senior Financial Analyst, and managed several financial departments at Ford from 1955-1965.

Mr. Roark was a Budget Examiner at the U.S Bureau of the Budget from 1947 to 1952. Previously, he was with the U.S. Weather Bureau and the Colorado Department of Revenue.

He is a graduate of the University of Colorado, where he received his BA (Magna Cum Laude) in 1940, and a Master of Public Administration degree in 1942.

He is past Vice President of the American Management Association, and was General Chairman of the 1979 National Conference of the Association for Computing Machinery.

1.1 THE FIRST TWO DATA BASE DIRECTIONS WORKSHOPS

Five years have gone by since the first Data Base Directions workshop in the Fall of 1975. That meeting was concerned about the fundamentals of database art--the language structures, the standards needed to govern future growth, and the benefits to be expected from the database environment.

Data Base Directions II, in late 1977, addressed a more advanced problem--that of conversion. Already it had become apparent, standards notwithstanding, that many of us would be faced sooner or later with the task of adjusting from one database environment to another. The value of these discussions was brought home to me two years later when there was an urgent phone call from a systems director of a large-scale government organization who said, "I've been told by my management to convert at once from batch to a state-of-the-art database environment. The proceedings of Data Base Directions II are the only literature I've been able to find on this problem."

1.2 DATA BASE DIRECTIONS III

Data Base Directions III has recognized a growing maturity and acceptance of database systems. In late 1980, most organizations were directing their major systems development toward on-line, database systems. The software market presented an outpouring not only of database management systems, but also of related products for the management of these systems and for the information resource as a whole. The evaluation of the resulting "Strategies and Tools" provided the theme for this third Data Base Directions workshop on October 20-22, 1980.

1.2.1 Using the Data Dictionary System.

The most obvious of these tools is the data dictionary system. Although most organizations have by now accepted the idea that information files ought to be organized independently of the computer programs they support, it is quite another thing to keep track of the contents of these files as they expand to encompass virtually every segment of information with which an organization is concerned. Most suppliers of database management systems have developed data dictionary systems to accompany their offerings. Several software firms have developed independent data dictionary systems. Most of us are still struggling to learn how best to use these tools for managing the rapid growth in our information resources.

1.2.2 Standards and Controls.

As always with the proliferation of new technologies, the explosion of information databases has led to a cry for "standards and controls." How can we apply standards and controls without stifling the growth we are trying to manage? This was a second theme of Data Base Directions III.

1.2.3 Database Design.

Finally, the growth of our databases has confronted us with serious problems of architecture. In the database field, we must consider two kinds of architecture--the "logical database design" and the "physical database design." Although these may be related, they are influenced by different needs and considerations. Data Base Directions III dealt with these problems through two additional study groups.

1.3 CONCLUSION

All three Data Base Directions workshops over this five year span have attempted to draw on the experience and skills of individuals who have achieved distinction either as practitioners or as students of database art. The proceedings of these discussions, therefore, provide an invaluable compendium of the accumulated experience of people who have been closely involved in the identification and solution of problems at the leading edge of database technology.

As a participant in these discussions, I found the individual workshops to be an invaluable stimulus to my own thinking and a rich source of ideas. All of us who participated in the meetings will hope that you, the reader, will share in the inspiration that we were able to gain through the three days of face-to-face meetings with our colleagues.

2. DATA: THE RAW MATERIAL OF A PAPER FACTORY

John A. Gosden

KEYNOTE SPEAKER

Biographical Sketch

John Gosden is the Vice President for Telecommunications for the Equitable Life Assurance Society, where he is responsible for planning, providing, and recommending management policy for all electronic communications, both voice and data. He joined Equitable in 1970 as a Second Vice President in charge of the Technical Support Group. From 1975 to 1978 he was in charge of Corporate Computer Services.

Mr. Gosden was with MITRE from 1966 to 1970. From 1966 to 1969 he was head of the Program Systems Design Subdepartment, and from 1969 to 1970 he was head of the Information Systems Department of the National Command and Control Systems Division of the MITRE Corporation's Washington office. Previously, he was with Auerbach Corporation and Leo Computers.

In 1977, Mr. Gosden was chairman of a Federal Advisory Group on White House Information Systems, which reported on ways to use information systems to improve the decision making processes of the President.

2.1 AN INTERESTING CHALLENGE

It is an interesting challenge to compose a keynote speech to such a specialized and expert audience. A dozen years ago I managed several projects and was very involved in databases. In the last ten years my experience has been with automation of a factory. An insurance company's home office is basically a paper factory, and the raw material is data. In the last twenty years insurance companies have automated the basic production lines and have pioneered many applications. Now we are looking at the future and there is much more to do, and one of the crucial technical problems is how we handle data: store it, index it, access it, update it. My speech today is essentially the content of a paper originally prepared for my top line managers. The material has not been modified for this technical audience. This is the way we described the next twenty years, how computers and communications will evolve, and how we expect business to react. I think it is an interesting commentary on today's insurance managers that such a paper with a strong technical background is an important part of their planning.

Section 2.2 discusses current changes in technology that will be familiar to you. The two systems mentioned, CAPS and EQUI-CLAIMS, are large, on-line, centralized database systems that support over half our labor and paper-intensive business activities. They cover over 3,500,000 individual life insurance policyholders and some 20,000,000 people covered for health insurance. These systems depend on databases. For these and many other applications we run, data dictionaries are a crucial tool.

Section 2.3 discusses a social and business phenomenon--instant service. We use a code phrase: "I want it now." Technology makes it possible. Business firms exploit it to get a competitive advantage, whether it is needed or not. Then it becomes a "want," and a commercial imperative. Thus, improved service becomes important competitively. This is complicated by an increasing volume of data that is made available by automation.

Section 2.4 predicts how managers and management must, and will, react to the greater service demands:

- Managers will become more analytical.
- "One-stop-processing" will be the dominant mode of processing.
- Service representatives will become key staff roles.
- Service representatives will be very demanding of their support systems.

Service representatives and managers will demand good data services, and these will be increasingly difficult to construct as the systems become wider-ranging and more complex.

None of this is wild imagination. All these trends are in place and happening today. The pace is determined by pragmatics, successful implementation, and the competitive situation.

The major point I want to establish and reinforce with you is that service considerations will drive database design much more than is now the case. We will want convenient access to a wide range of data. It will need to be linked together and cross-referenced in a number of ways, and these ways may be continually modified. This indeed is a very difficult problem, and we will be waiting for your solutions.

2.2 CHANGES OCCURRING IN TECHNOLOGY

Over the last thirty years there have been dramatic changes in computing and communications technology. The computer has evolved as a powerful tool, and has been increasing in power more than tenfold each decade. We now have a wide array of computers: large, medium, mini, and minute--even pocket calculators that are more powerful than the large computers of 20 years ago. Communication technology has also grown dramatically. We now have large networks of terminals and computers. We have changes in telephone systems every few years instead of a very stable and stagnant set of systems. Communications and computers have made possible large systems such as CAPS and EQUI-CLAIMS and have provided a basis for regionalizing service while maintaining a common system and centralized databases.

Where is technology going in the future and how will it change our way of doing business? We expect dramatic changes to continue in the next twenty years. The major changes that we will see will occur because systems and services can be larger, more comprehensive, faster (or more responsive), and all at lower costs. There are more than enough new ideas around, and as costs come down new ideas become economic and attractive. What follows are some developments that I expect to become available.

2.2.1 Speech Filing.

Many simple telephone messages will be automated. Today, people leave messages to call each other back; this causes frustrations and delays. Soon we will have systems with which we will be able to dictate messages to each other and call up on our own telephones to listen to the messages dictated for us. The messages can be stored, and even forwarded to others. Any new message can be automatically sent to a list of people. If you are away from your telephone you can dial in to get your messages.

2.2.2 Electronic Mail and Storage.

Messages, memos, letters, and contracts that need to be in writing will appear only rarely on paper. They will be prepared on terminals, sent electronically over networks, read from a screen, indexed and filed electronically. Managers and staff will be able to use terminals to scan and retrieve documents needed to review history, or to record answers to questions. Paper and copying costs (dollars and wasted time) will be cut (drastically, I hope), mailing delays avoided, and many problems of keeping files up to date and finding things in them should be much simplified.

2.2.3 Image Storage.

Some important documents, photographs, signatures, and letters that arrive in non-electronic form will have to be stored. Today, these are filed because they are too expensive to store in electronic form. With reduced costs we will be able to afford to supplement automated electronic filing systems and keep such documents as "photographic" images in our electronic files, thus eliminating more paper, and saving the problems of transcribing them.

2.2.4 On-Line Reports, Analysis, Graphics.

The new systems cannot simply present raw data to executives. Just having all the operating data on-line does not mean a manager can just press some buttons and see a useful management report. Actual results that are different from predictions or plans need interpretation. Experts will have to analyze and filter results, look for causes, and separate the effects of each. They will use powerful online analytic tools and create management reports using graphics technology to produce diagrams, tables, charts, and graphs to display results. Where models have been developed, new trends can be fed into them to project possible changes and allow management to consider new alternatives.

2.2.5 Retail Databases.

Today, there are some databases maintained and available for a fee: Stock Exchange prices and volumes; indexed legal rulings; travel and vacation data; industry statistics; and several government statistical publications. More of these services will be available and used by managers. Private exchanges may well develop, and we may also expect in-house databases to be organized so that they are useful to other parts of the organization. For example, both the personnel office and the controller need to look at employee data.

2.2.6 Variety and Convenience of Access.

Small terminals will be available for use at home to access business systems. Today, the special communications lines needed are expensive and take time to install. By 2000, cable TV systems will allow home terminals to be connected to company systems without having to install special lines, and without being restricted by the limited capacity of telephone lines.

2.2.7 Cable TV Extensions.

Sophisticated cable TV, including systems such as "viewdata," will become available. Viewdata is a new type of service that enables suppliers to make statements available about their service and products. Viewers can tune in, scan for products they are interested in, see a telephone number, call up, and conduct business. Many simple functions will be carried out by a simple keyboard, similar to one on a touch-tone phone, and the TV screen will be used for replies. In short, a large variety of devices and services will be developed around the home market.

2.2.8 One-Stop Processing.

One-stop processing is a term used to describe a system in which a single person can provide a complete range of services. Thus, a customer can be dealt with by one person without having to go to various desks or counters to be serviced. A good example is an airline reservation system where one clerk can make all the reservations, check credit, issue the ticket, and check baggage. Such systems depend on good on-line databases, relatively cheap terminals, cheap processing, voluminous storage, and, most of all, the ability to design and install such systems even though they require dramatic changes in organizations and in employee attitudes.

2.3 GENERAL FORCES STIMULATING CHANGE

Working alongside improvements in technology are consumerism, changes in customers' expectations, and some current frustrations that are stimulating changes in society and business.

First, there is a pervasive theme emerging today that applies to customers, employees, managers--indeed a pervasive theme in our society. I believe it will also be a characteristic of the early twenty-first century. It is expressed in the phrase I want it now! A generation of people are growing up in an environment where they write few letters but make extensive use of the telephone. They expect to be able to get a response or action right away--buy something with a credit card, make a reservation, arrange an appointment, get a prescription from a doctor.

Second, competitive forces will continue to be a strong driving force, and service orientation may become more important than price (just as with convenience foods). Once service becomes that important, all vendors must adopt such an attitude or face a declining business.

Third, there is the problem of coping with an increasing volume of data. Today, government and business organizations are drowning in data. At the same time, the computing and communications technologies are increasing rapidly in scope and power. Information technology, the techniques by which the tools are used for the interpretation, presentation, and controlled use of data, are in the early stages of development. The current substantial investment in "Office of the Future" projects by American business, however, reflects a growing interest in this area.

Overall, in the next twenty years I expect that technology, business, and social pressures will accelerate development of systems that respond faster to the needs of business managers, staff, and customers.

24

2.4 THE NEW MANAGER'S WORLD OF 2000

There is no single picture of the future--any specific statement is only a prediction--but we can predict a continuance of a general trend that already exists. There will not be one general pattern. Different industries, different organizations, different managers will have different needs and different styles, just as they do today. That is why we can give a general picture with confidence but specific examples are less reliable.

Much of a manager's time is involved in dealing with messages (telephone calls and memos to be answered), studying data (operating results, status of projects), making decisions (looking at alternatives and weighing possible outcomes), and conducting or attending meetings.

A manager will use on-line systems extensively--some directly for messages, some for looking at reports, and some for following up. His or her staff will use on-line systems to review results, to consider alternatives, and to develop plans. Operating staff will use on-line systems to perform the basic functions of a business.

Good managers will emerge who will operate less by "the seat of their pants" or intuition, and will rely more on evidence. They will set up systems that produce operating data that show what is happening, use models to consider alternatives, and be more analytically oriented than today. It will be possible to make choices much less of a gamble, to expect a faster awareness of, and reaction to, changes. Managers, too, will become impatient. They, too, will "want it now."

A new management philosophy and an increasing tempo of business will require managers to make decisions more rapidly. At the same time, with more information readily available, it will be possible to increase the span of control of an individual manager. There should be fewer people to deal with exceptions or to supplement the automated systems and cover the connecting parts. Most of the processing can be automated. The major outstanding functions will be analysis, decisions, and customer service, as well as planning. Thus, there will be fewer pure administrators and "bean counters," and more managers with the entrepreneurial outlook to take advantage of emerging opportunities. These new managers will have a broader scope of action, will be more "risk" oriented, and will be held more directly accountable for results.

Organizations need to change to support one-stop processing; previous organization structures divided work up so that separate units specialized in specific activities, and a transaction would move through various work units or work stations. In my organization, for example, when we want to deal with a caller on-line, whether at a counter or on the phone, we must arrange things so that one person on our staff can deal with all the questions and actions the caller wants. We cannot provide service, nor be efficient, if we pass callers from one unit to another.

Employee attitudes need to change because there is an entirely different tempo and inter-personal style required to respond directly to callers rather than processing pieces of paper. The tempo is partially controlled by the caller and work cannot be rearranged into neat piles of similar kind. There is no time to "send for files"--files must be ready at hand. There is no time to refer unusual cases to a supervisor--too many exceptions will stop the system from working.

The tempo of both society and business will be affected. It will be a "dynamic society" willing, able, and wanting to do it now.

Dedicated employees wanting to provide good service can be expected to demand that the system and organization support them in their jobs. They will be critical and intolerant of inefficient or unresponsive systems, and of inefficient managers who do not fix things or develop ways to respond to new situations. Again, a very dynamic environment is going to emerge. As employees are more involved in direct service, management must be more responsive and less autocratic. To get responsive employees, managers will be called upon to explain what is going on, and truly get employees involved in the business, because, as the service interface, the employees will be a major part of what the public sees as the business. Not only will there be a need for more delegation within management to mini and mini-mini businesses, but many employees will need to be considered as managers of their own time and company resources, as they operate directly with the public.

Where will more responsive employees come from? They are already becoming available. The younger generations in our society seem to be more attuned to "people-needs," and thus should be sympathetic to a service attitude.

In the next 20 years I expect the combination of the available technology, the already established advantages of one-stop processing, management frustrations with delays in getting data, and a continuing demand for service to foster an "I want it now" climate.

3. USES OF THE INFORMATION RESOURCE DICTIONARY SYSTEM FOR IRM

Anthony J. Winkler

CHAIRMAN

Biographical Sketch

Anthony J. Winkler is a Senior Consultant with Alpha Omega Group, Inc. and Chairman of the American National Standards Institute's Technical Committee on Information Resource Dictionary Systems. Dr. Winkler has over 20 years experience in computer related areas as manager, researcher, nationally known speaker, and author in the subject areas of implementing resource management and using dictionary systems as management tools. He was Chairman of the Air Force's System Survey Team which published *The Data Administrator's Handbook*, the foundation document for database administration procedures within the Worldwide Military Command and Control System (WWMCCS).

PARTICIPANTS

- Carl Clark
- Al Dale
- Charles Drumm
- Don Fox
- Gayle Gillingham
- Bob Hegland
- Allan LaRue
- Tom Lee
- Marv Lerfald
- Dan Magraw
- John Mayson
- Don McCaffrey
- Has Patel
- Susan Rosenbaum
- Roy Saltman
- Dan Schneider
- Jim Swager

3.1 INTRODUCTION

3.1.1 The Objective.

The objective of this panel was to identify the uses of a data dictionary system in supporting organizations which have recognized the need for and are attempting to implement an information resource management (IRM) environment. Although a data dictionary system, in its most rudimentary sense or its normally understood sense, can support some aspects of the IRM environment, the requirements for IRM go beyond the functions of most if not all currently available dictionary systems. For this reason it is the consensus of this panel that the expression "data dictionary system" is no longer appropriate in describing a prime tool of IRM. The expression chosen by the panel is the "information resource dictionary system" (IRDS). This expression was chosen because it is inherently more descriptive of the purpose of this tool and is more closely in tune with the trend of dictionary systems in the market place.

Before proceeding, it is necessary to explain the panel's intended meaning of IRM and IRDS. The principal points in these definitions concern the broad coverage of information and information's relevance to and availability for management needs across the enterprise.

- Information Resource Management (IRM) is whatever policy, action, or procedure concerning information (both automated and non-automated supported) which management establishes to serve the overall current and future needs of the enterprise. Such policies, etc., would include considerations of availability, timeliness, accuracy, integrity, privacy, security, auditability, ownership, use, and cost effectiveness.

- An Information Resource Dictionary System (IRDS) is an information system with automated support which documents the information environment of an enterprise, supports the operational aspects of that information environment, illustrates the interrelationships of information environment components, and documents the locations of all components of the information environment. The Information Resource Dictionary (IRD) is the actual database manipulated by the IRDS software.

IRM is currently one of the most significant topics being discussed concerning information systems, and is being discussed along a variety of lines of thought. These include business systems planning; information systems analysis, design, and development; database design and implementation; the disciplines of office management, paperwork management, and information sciences management; and the various problems and costs associated with implementing IRM to include each of these areas.

The panel would like to point out that IRM and an IRDS are not panaceas. They, by themselves, will not eliminate all of the problems encountered in the previously mentioned areas. They can support the resolution or elimination of these problems, but only if the goal of the enterprise is to manage the resources which provide it with information. In this case, all components of the enterprise's information environment must be identified and documented to the level which makes the components "shareable" across the enterprise. To realize this goal, IRM is a necessary function and the IRDS is a necessary tool of IRM. How the IRDS can be used to support this goal will be discussed in later sections of this panel's report.

3.1.2 A Historical Perspective.

Terminology changes since Data Base Directions (DBD) I (1975) and II (1977) are revealing. The emphasis in these earlier workshops was "data"; today it is "information resource." The emphasis on "data management" discussed in the keynote speech at DBD II has evolved to "information resource management" at DBD III. These changes are, of course, more than word changes. They reflect an increasingly prevalent view that the narrow concern for how computers manipulate data has outlived its usefulness. In is place is a global view which is far beyond manipulating data within an automated database; the view is concerned with providing all information in the form and at the time required by the enterprise. In fact, since it is frequently postulated that an enterprise's greatest asset may be its information resource, management of this resource is as necessary as management of the enterprise's personnel and financial resources.

Management of the information resource is a developing art. Its primary objective is to meet, with increasingly effective results, the needs of the enterprise for information. This requires identification, review, continual updating of needs, and development of information systems which meet those needs with either internally generated or externally available data. This further requires a framework for the enterprise and thus necessitates the establishment of plans and policies to assure consistency of information systems with enterprise objectives, plans, priorities, legal constraints (such as data privacy), and security.

Implementation of the IRM concept necessitates the creation of an information system which might be called an information resource management system. This system, in order to be effective, must permeate the enterprise if it is to assure the integrity of the enterprise's information environment. Such an information system, to be more than a short-lived experiment, must be adequately staffed and funded, and must be supportable with sound cost/benefit analyses. Both short-term and long-range costs and benefits--of all kinds--must be continually projected and monitored. Over time, the enterprise's objectives, priorities, resources, and information requirements, its available technology, the ability of its managers to use information, and the cost/benefit trade-offs associated with information use will change. What will not change is the need for an optimal mix of information system resources, and the need for management at all levels to assure that the right information is available at the right time to aid in problem solving and decision-making for the enterprise.

3.1.3 The Approach.

The problem facing this panel was to illustrate how the IRDS could be used to support managed information environments to realize the concepts and benefits described above. The approach taken was to discuss the information environment from three different perspectives:

1. the enterprise's organizational perspective
2. the perspective of information systems supporting the enterprise
3. the perspective of the enterprise's database, which is the source of all data used by its information systems to produce desired enterprise information

The organizational perspective section identifies three classes of users:

1. functional managers
2. information resource managers

3. information consultants

Four IRDS metadata classes are required to support these users:

1. data and information class
2. source class
3. use and user class
4. process and handling class

A discussion of the uses of the entity-types within these classes is also presented.

The information system approach was to identify the users, uses, and types of use of an IRDS for each phase of the system life cycle. The phases discussed in this section are:

1. the requirements definition phase
2. the development phase (including analysis, design, and programming and testing)
3. the installation/integration/conversion/testing phase
4. the operational phase
5. the maintenance and modification phase

Tables describing the interrelationship of users, uses, types of use, and phases of the system life cycle are presented as an attachment to this chapter, in Tables 3-1 through 3-9.

The database perspective approach resulted in the definition of a logical IRDS consisting of an enterprise-oriented dictionary function, a directory function, and a system-specific dictionary function. The users and uses of each of these functions is described in terms of an enterprise's database--whether manual or automated or both. Each perspective will be discussed in detail in the remainder of this chapter.

3.2 THE ORGANIZATIONAL PERSPECTIVE

3.2.1 Introduction.

Much of what has been written and spoken about IRM emphasizes:

- IRM related technology, both hardware and software
- the implications of IRM on information systems and automated database design

What is too often left unsaid or lost from sight is that IRM's reason for being is to serve the information and decision-making needs of the enterprise. In the words of the keynote speaker of Data Base Directions I in 1975: "The essence of management . . . is rational decision-making based on the best available data. Thus, the broad questions of data management lie at the very center of the management process." Extending this more specifically to the topic of this workshop, the IRDS, IRM and all related policies and procedures must evolve to serve the total information and decision-making needs of the enterprise.

3.2.2 The Organization, the IRDS and IRM.

IRM is an enterprise-wide management program which is global in its objectives and scope. Like all management programs that extend across all organizational components of an enterprise, an IRM program must strive for information consistency, completeness, and compatibility. That is, the information resources of Division A must be defined, described, and noted in the IRD (the actual database of the IRDS) in the same way as are the information resources of Department X.

The goal of consistency has several implications for an IRDS and the information system of which it is a part. Foremost among these is the need for the IRDS to support a "top-down" approach. This top-down approach should reflect the enterprise's organizational structure and, perhaps more importantly, the enterprise plan, enterprise model, or long-range enterprise strategy, and its association with the information systems which support it, as well as the components of each information system. A significant feature of the top-down approach is that it has no inherent limitation of depth, and will thus allow the IRDS to support organizational and information components at all levels of the enterprise hierarchy, as well as a global enterprise view.

The needs and realities of large enterprises may require them to implement multiple IRDSs to support the operational aspects of IRM. All that IRM demands with respect to such multiple implementations is consistency of the IRD contents across the enterprise. This consistency may be best accomplished by treating all IRDS as part of a global, logical IRD; the database section of this chapter proposes such a model for the IRDS.

3.2.3 Organizational Users of an IRDS.

What is now happening in the IRM environment is an elevation of the IRDS concept to the enterprise management level. In this arena the IRDS is used to manipulate not only data about data, but also data about the information environment including specific names, places, functions, etc., by which the enterprise's information holdings are linked to the structure, functions, clientele, etc., of the enterprise itself. Furthermore, if the IRD is to be truly enterprise-wide in is scope, it must contain references to both internally generated information and information acquired from external sources.

The IRDS should provide information about those enterprise information resources required to support management needs at all organizational levels. These needs exist both for those managers directing the business of the enterprise and for those managing its information resources. In addition, the IRDS must support those people within the enterprise who bridge the gap between the end-users of the information and the developers and providers of the information base.

The panel identified these user classes as functional managers, information resource managers, and information consultants. A description of each class and its potential uses of the IRDS is provided below.

* Functional Managers. The functional managers are charged with executing the business plan of the enterprise; to accomplish this they need an inventory of information available to them to assist their decision-making. These managers, and their staffs, will use the IRDS to provide a catalog of available information. The IRDS can give a perception of what is available, the form in which it is available, where and how to obtain it, and its probable cost. The IRDS should help these users scale their information requirements to only what is involved in satisfying those requirements and can thus contribute directly to controlling information expenditures.

For managers engaged in planning for the enterprise, the IRDS can help to identify proper goals and objectives. Controllers and auditors, on the other hand, can use the IRDS to verify goal accomplishment and enterprise performance; they will probably be among the most active and demanding of users.

- Information Resource Managers. These are people who are concerned with the cost-effectiveness of those components of the information environment used to generate, receive, transmit, process, and store information. Included in this class are those responsible for data collection activities, data administration functions, libraries, telecommunications, office systems, and applications systems design, development, and operation.

  These managers will look to the IRDS for analytical data on how often and how effectively the information resources are being used. The IRDS should also provide them with the information needed to analyze and assess the impact of proposed changes to an information resource. Information resource managers will also use the IRDS to accomplish special functional management requirements such as those concerning security and/or privacy issues.

- Information "Consultants". Information consultants are those intermediaries who, like reference librarians, can communicate with information consumers in the consumer's language(s), can help the consumers articulate their needs and frame their inquiries, and can locate the information required to satisfy the consumer's requests.

  Information consultants will find the IRD an indispensable tool, since it unites data about all enterprise information resources. They will use the IRDS to learn about both the existence of desired information and its location. In this respect, the IRD contents identify the total information holdings of an enterprise much as the contents of a card catalog identify the books, periodicals, etc. of a library.

  Requests for information may be very general or very specific. For example, a state legislator may inquire of the commissioner of education, "can you give me a breakdown for the elementary schools in my district of enrollment figures by sex within grade, for each school?" The IRDS would be used by an information consultant to determine if there is an information source which contains the desired information, or if the data exists in a form which can be processed to produce the desired information. The IRDS would not be used to provide the desired "breakdown" itself.

3.2.4 Scope of the Information Resource Entity-Types.

The contents of the IRD is essentially a well-defined collection of occurrences of entity-types (information environment components), their associated attributes, and a representation of relationships among the entity-types. To satisfy the IRM needs of the enterprise, the IRD must contain at least the following four classes of entity-types:

1. The data and information class includes those entity-types that describe data about data in both its "raw" form and its processed form. Included among these entity-types are databases, files, reports, forms, records, and items.

2. The source class of entity-types describes the origins of data and information. These may be customers or clients, publishers, government, business processes (e.g., a manufacturing or sales activity), or even employees.

3. The use and user class of entity-types are essentially of two types--enterprise functions and components. Enterprise functions (e.g., manufacturing, sales, accounting) reflect the functions supporting the plan or program of the enterprise. Enterprise components describe the discrete organizational units in which functions are performed.

4. The process or handling class of entity-types cover the breadth and depth of information handling in the enterprise. They are not limited to the computer related entity-types such as automated applications, programs, modules, etc., but also include the information systems (whether manual or supported by computers) of the enterprise and the tailored procedures used by enterprise personnel to provide the desired information. These information systems include those used within libraries, record repositories, and document handling centers.

Within each class, the entity-types should include all attribute-types required to fully describe the entity-type and to use the entity-types and their attribute-types in an operational sense. The representation of relationships among these entity-types serves to "link" data and information entity-types with their source, use, and handling entity-types. Furthermore, there should be relationships between sources and uses, sources and processes, and uses and processes.

The concept of the IRDS discussed above leads to the realization that the contents of an IRD is actually a management database and the IRDS is, in fact, a management information system that can be used to determine: who knows what, who does what, how information systems fit together, where information comes from, and, potentially, why information is created.

3.3 THE INFORMATION SYSTEMS PERSPECTIVE

3.3.1 Introduction.

This section discusses potential uses of the IRDS during the development of an information system. An information system development begins with an enterprise problem stated as a requirement. This requirement is then translated into procedures which satisfy the requirement. During this evolution the IRDS is used as a support tool. To relate the users of the IRDS and the uses made of the IRDS, the evolution is discussed herein in the framework of an information system's life cycle (SLC). The IRDS can be extremely useful during the SLC as a tool in documenting, maintaining, and controlling an information system's analysis, design, and development, and in facilitating its operation.

By using the phases of an information system life cycle, it is easier to specify the requirements for the IRDS at each phase and to specify the interfaces necessary to tie all the IRDS views together into a coherent and minimally redundant information system. The approach, therefore, provides a framework for determining and understanding the relationship of the IRDS to enterprise's information systems.

3.3.2 Definition of the System Life Cycle.

Since there is no standard information system life cycle, the following discussion is presented to define, for this report, the SLC phases and their uses. Figure 3-1 provides a pictorial representation of the SLC.

1. Requirements Definition Phase. This is a response to a customer request for an enhancement to an existing production information system or a request for the development of an entirely new information system. The output of this

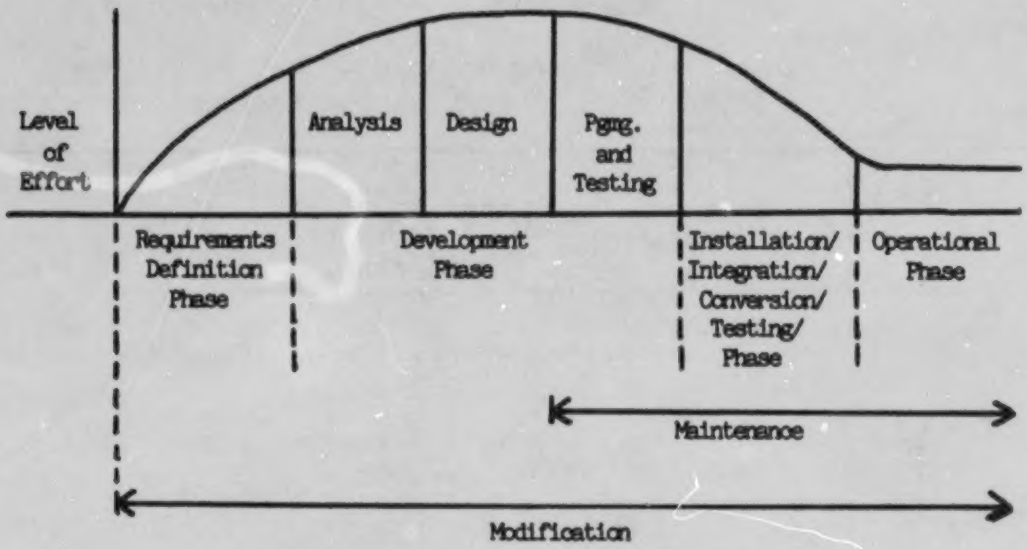
Level of Effort
Analysis
Design
Pgmg. and Testing
Requirements Definition Phase
Development Phase
Installation/ Integration/ Conversion/ Testing/ Phase
Operational Phase
Maintenance
Modification

Figure 3-1. Information System Life Cycle

phase would be a preliminary proposal whose key components are:

- a "rough cut" view of how the system would work
- hardware/software performance and standards requirements
- data resources needed
- staff resources needed
- impact on other system components
- computer resource requirements (storage and processing time)
- benefits (positive and negative; tangible and intangible)
- summary justification to continue or discontinue with the development phase

2. Development Phase.

A. Analysis. The analysis stage is the formal effort in response to a customer request for an information system enhancement or development effort, including:

* complete analysis of the request
* interview of customers and other affected users, and documentation of the current information system
* clarification and acceptance of objectives
* determination of output, input, and procedures required
* development and presentation of alternate solutions
* selection of the best solution
* information system flowcharting
* approval and user acceptance of a solution
* establishment of project schedule and selection of staff
* final pre-design approval and sign-off

B. Design. The design stage is the development of the specifications for the information system, software, and operations including:

* information system: summary flowcharts and narrative
* information system: mid-level flowcharts and specifications
* manual and automated procedures: summary flowcharts and specifications
* program summary: flowcharts and narrative summary
* program: detailed flowcharts and detailed specifications
* operations flow, job control and scheduling: external balancing and distribution specifications
* quality assurance approval and sign-off

C. Programming and Testing. Activities during this stage include the efforts of the programming (coding) personnel. Included are:

* coding in the selected language
* sorts, merges, and other utilities
* execution/job control language
* job stream structures
* compiles
* debugging
* program testing with programmer test data
* approval and sign-off

3. Installation/Integration/Conversion/Testing Phase. During this phase of the life cycle the following activities take place:

- information system testing including formal test data and parallel testing
- conversion of affected files and automated/manual procedures
- integration into the data processing and corporate processing cycles
- implementation or installation into a distributed and/or centralized location according to an approved implementation schedule
- quality assurance approval and sign-off
- customer approval and sign-off

4. Operational Phase. The activities during this phase are directed towards the operations assurance effort and include:

- review of job schedule, priorities, runtimes, etc.
- review of external balance procedures, distribution, etc.
- review of hardware requirements
- review of retentions, back-up and recovery, disaster recovery
- review of ownership and trouble call assistance
- quality assurance sign-off that all operations standards are being met

5. Maintenance and Modification Phase. The activities performed during this phase include:

- the continued maintenance of a production information system to keep that system in line with department standards and changing hardware/software configuration, etc.
- the implementation of modification requests to change, add, or delete the processing specifications of an existing production system. A new project life cycle or parts thereof, depending on severity, will be initiated to effect such modification.

The activities described above for each phase of the system life cycle provide a framework for comparing uses of the IRDS. Before showing this relationship, it is necessary to also identify and clarify the potential users of the IRDS. We will then illustrate the relationship between these users and their uses of the IRDS during the SLC.

3.3.3 Using the IRDS to Support Information Systems Evolution.

The SLC provides a framework for determining the relationship of the IRDS to information systems. In general, the IRDS would be used to:

- provide information about information systems for data contained within the IRDS
- report on existing conditions and environments
- support the information system evolution throughout the system life cycle
- support "what if?" investigations and analyses

Who the IRDS provides support to and what kind of support it provides is expressed in the following sections. This discussion should illustrate the wide variety of users and uses of the IRDS during the SLC phases and activities. It is hoped that Tables 3-1 through 3-7, used to illustrate these relationships, will clarify an area which is normally very ambiguous. The tables are presented as an attachment at the end of this chapter.

3.3.4 Users of the IRDS.

The potential and actual users of the IRDS in the development of information systems can be individuals, but the intent here is to consider them as classes of users or as organizational entities. The following items describe the identity of each user and the functions of these users in the various phases of an information system life cycle:

- The Information Resource Manager is responsible for managing the corporate information resource. These resources include but are not limited to:

    - manual data and information
    - automated data and information
    - voice, text, and image data and information
    - formal corporate information and its flow
    - corporate records and information archiving
    - procedures and forms
    - corporate information systems

- The System Designer is responsible for the development of information system design (information flow and processing, man/machine interfaces, etc.) specifications.

- The Programmer is responsible for developing the computer programs using procedural languages (COBOL, PL/I, etc.), non-procedural languages (report writers, file management systems, etc.), and program/system generators (input validators, application development tools, etc.).

- The Customer is a requester and/or end-user of the information system's products.

- The Operator is responsible for the efficient scheduling and operation of the computer-related equipment (computers, printers, COM, etc.).

- The Project Manager is responsible for managing the different information system projects from development analysis through the implementation phases of the life cycle. In some cases, project managers are computer systems analysts; in others, they are lead individuals from the customer areas.

- The Maintainer is a programmer or analyst responsible for the information system while it is in the maintenance and modification phase of its life cycle.

- The Database Administrator is the individual responsible for the integrity of the physical data structures and definitions, and for database performance.

- The Data Administrator is the individual responsible for the integrity of the logical data structures and definitions.

- The Systems Analyst is responsible for analyzing user needs, information flow and transformation, privacy and security, information processing, and other information system requirements.

- The Auditor is responsible for analyzing audit and control information during the various phases of the system development life cycle. These individuals may include auditors, quality assurance analysts, and customers.

- Programs are automated procedures that may directly access the dictionary for dynamic use of the metadata.

3.3.5 Uses of the IRDS.

In identifying the potential uses of the IRDS, it was clear that terminology in this area is inconsistent. The following explanation is provided to clarify the terms used:

- Techniques and processes describe algorithms and control processes.

- Source methodologies and justification describe generic business applications.

- Networks describe the relationships between systems components and data, including their dependencies--both implicit and explicit.

- Data flows depict the flow of data through the system and identify processes and procedures affected by the flows and processes and procedures affecting the flows.

- Data structures depict the characteristics of data and data access techniques.

- Integrity describes business, legal, application, etc., constraints, including security and privacy requirements to insure correct aspects and consistency of data.

- Validation criteria describe legal and/or illegal domains of values.

- Data definition generation describes the information necessary to map data from the business view to the automated view, e.g., skeleton for generating DDL, PL/I declarations, subschemas, etc.

- Impact analysis portrays the information necessary to assess the impact of changes to the information system because of modification to the business environment, resource requirements and utilization, system and component interfacing, the user community, etc.

- Information system design provides information to assist the definition, design, and integrity of the total system.

- Backup and recovery identifies requirements and constraints to perform system backup and recovery.

- Environmental constraints define the factors which will constrain either the development or the operation of an information system.

- Monitor application system use is the use of the IRDS to actively maintain information on how the information system is being used and who is using it.

- Conversion defines those factors to be considered when an information system is being converted from one environment to another.

- Metadata changes identify the owner or focal-point for IRDS information to insure change is controlled.

- Forms design defines and describes all formats for input and output.

- Transformations define codes (or values) with their associated meaning or text.

- Program code generation requires the documentation of data definitions, data flow, processing techniques, forms design, algorithms, and networks to create program source for an application development facility.

- Test data generation requires the documentation and analysis of data definitions, data flow, processing techniques, forms design, algorithms, and networks to create test databases, test transactions, test plans, and test reports.

- Versions define multiple versions of the same information.

Tables 3-1 through 3-7 illustrate how the IRDS is used in each SLC phase by identifying the type of use for the users and uses described above.

3.4 THE DATABASE PERSPECTIVE

3.4.1 Approach.

The objective of this section is to conceptualize an IRDS in relation to the design and development of databases, querying and deriving information concerning databases, and providing operational support to automated databases. This database perspective is also equally applicable to non-automated files and operations. The terminology in this section should therefore be interpreted to refer to both the automated and non-automated environments.

3.4.2 Users and Uses of the IRDS.

This approach recognizes three conceptual functions in terms of two modes--global and operational. The specific needs of each mode are significantly divergent and are therefore addressed individually. The global mode incorporates an enterprise-oriented dictionary function (Figure 3-2) and is thought of from the enterprise point of view, since all data about the enterprise can be viewed as being independent of environment (manual or automated), physical implementation, computer systems, or hardware.

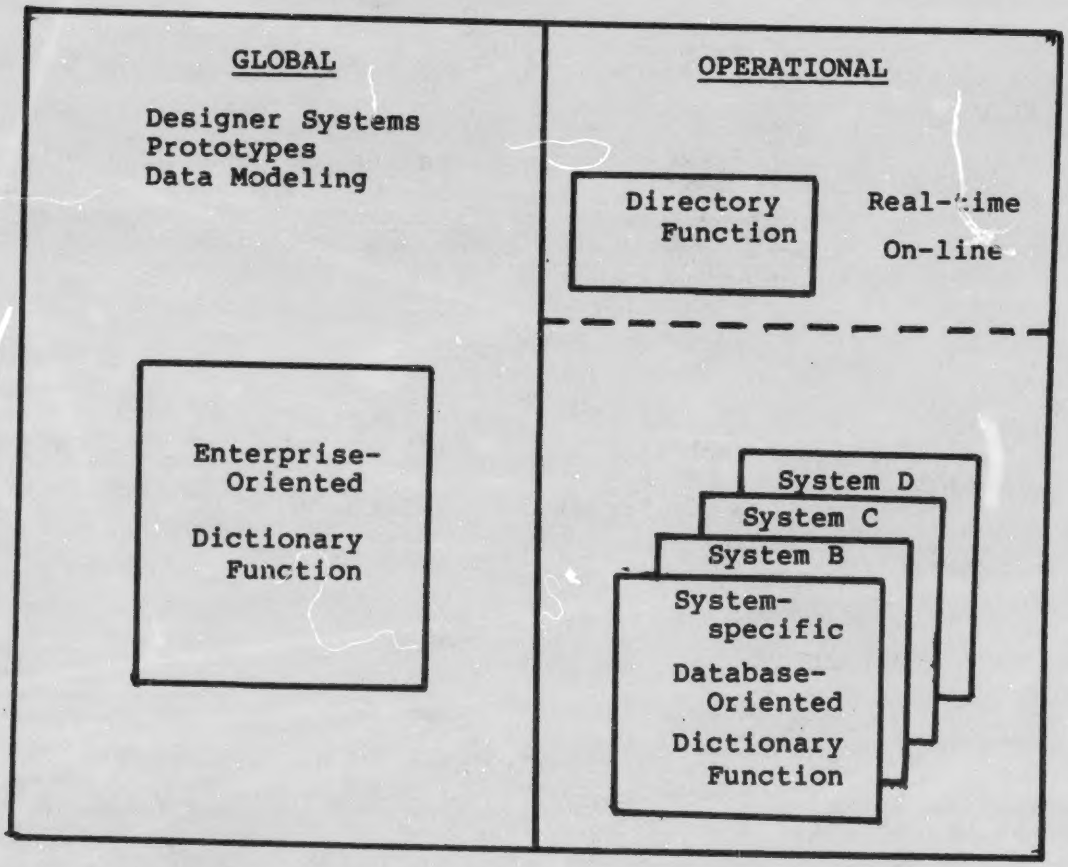
GLOBAL
Designer Systems
Prototypes
Data Modeling
Enterprise-Oriented Dictionary Function
OPERATIONAL
Directory Function
Real-time
On-line
System D
System C
System B
System-specific Database-Oriented Dictionary Function

Figure 3-2. Database Perspectives of an IRDS

The operational mode, on the other hand, is composed of a directory function and a systems specific dictionary function (Figure 3-2). The directory view concerns user inquiry, information searching, and information locating in terms of the data and information available in the operational data environment of the enterprise. This concept is designed to accommodate a multi-computer system, multi-database, and multi-location (distributed) information environment. The systems specific dictionary is a database oriented dictionary view concerning the operational needs for an IRDS for each specific data processing environment. It assumes that each hardware system may have a distinct dictionary capability, each database management system (DBMS) may have a distinct dictionary capability, and several "stand-alone" dictionary capabilities may exist on several hardware configurations. Although three conceptual functions have been identified, they may in fact be coalesced for enterprises within a diverse hardware or software environment. From a logical perspective it is important to view these three conceptual functions as belonging to a single, logical IRDS.

This conceptual approach makes clear the distinction between data administrator and database administrator activities. The data administrator activity must be concerned with the enterprise view of the dictionary and looking at data and information in global terms. The database administrator requirements are specific to each particular database environment as defined by the physical implementation within the enterprise, and will be concerned with the physical and logical aspects of the operational information environment, which is usually dependent upon the size and scope of the enterprise.

3.4.3 Enterprise-Oriented Dictionary Function.

In the context of the database environment of the enterprise, the enterprise-oriented dictionary function:

* provides a canonical description of all data objects and associations among data objects which may be referenced or utilized in databases implemented in the DBMS environment(s) available to the enterprise. Such description will include authorized synonyms of named data objects and associations.

* identifies aggregations of data objects and associations, as necessary, consistent with the data model used to structure the (conceptual) global database schema, and thus defines the conceptual schema

* identifies the use of data objects in user views (external schemas) implemented in the DBMS environment, and the computer system environment(s) in which user views are implemented

- identifies semantic integrity constraints, access controls, and validation criteria with respect to data objects or aggregations of data objects defined in the conceptual system

- supports the following functions, when embedded in appropriate software environments:

  - conceptual (global) database design--this includes provisions for merging user views into a non-redundant global view if the design methodology is requirements oriented.
  - mapping user views at the conceptual level to external schemas in the operational DBMS environment(s)
  - mapping the systems view to the directory in the operational DBMS environment

3.4.4 Directory Function.

The directory is an organized, integrated repository of data that provides for user inquiry, information searching, and information locating to determine what data and information is available in the operational data environment of the organization. A functional user, regardless of discipline, would be able to determine whether the data exists in the enterprise and, if it does exist, where it is located and the path to obtain the information. The directory function provides a cross-reference use of data whether it is involved with the interrelationship of the hardware and software such as for multi-systems, multi-databases, multi-locations (distributed), different hardware configurations, and different database management systems; or the interrelationships of data such as user views and synonyms.

Auditors and functional users would be the primary users of the directory. The auditors would use the directory to audit the functions and uses of the enterprise-oriented dictionary, whereas the functional users would use the directory to reach his or her system specific dictionary. In addition, the directory could act as a controller or buffer to the user when one of the system specific dictionaries was not on-line. This would provide information to the user to allow him or her to make a decision to either wait until the system specific dictionary comes back on-line or access an alternate source for obtaining the appropriate information.

3.4.5 System Specific Dictionary Function.

This function concerns the support to be provided for a given DBMS. It can be divided into the following subfunctions which are further divided into the major IRDS facilities required to implement these functions:

* Describing databases
  - defining physical structures
  - defining relationships between physical structures
* Database version control
  - providing staging and status facilities
  - providing reorganization criteria and data
* Monitoring and analyzing use
  - collecting frequency of access and update statistics
  - determining processing time for access and update
  - tracking resource usage (central processor, input/output, etc.)
  - collecting failure statistics
* Generating schema and subschema definitions
  - accessing logical structures from the enterprise-oriented dictionary function
  - generating schema definitions
  - generating subschema definitions
  - generating input/output control parameters for the application programs accessing the DBMS
* Implementing integrity
  - storing the integrity constraint rules
  - applying these rules when new databases are generated
  - randomly checking to insure that integrity rules are not violated
* Access constraints
  - sorting information related to access and update constraints for given parts of the databases
  - applying these constraints when access and update requests are processed by the DBMS

- **Validation criteria**
  - accessing validation rules from the enterprise-oriented dictionary function
  - converting these rules into the DBMS specific format (e.g., internal tables)
  - applying these rules when update requests are received by the DBMS

- **Encode/decode**
  - encoding any user-specified information (e.g., schema and subschema definitions) into internal working formats and tables
  - decoding internal working format and table information into user-specified format
  - allowing the user to update the decoded information
  - generating necessary error and warning messages during the encode/decode process

Tables 3-8 and 3-9 provide, respectively, a tabular view of the relationships between the users and the three logical IRDS functions, and the uses of these IRDS functions from the database perspective.

3.5 CONCLUSION

The reader of this report should be convinced that his or her enterprise needs a managed information environment. This realization may have resulted from continual exposure to the ideas of IRM, IRDS, data management, database administration, configuration management, quality assurance, productivity, maintainability, etc. or it may have been an inspiration. In either case, this chapter of the report has provided a model for uses of an IRDS. All aspects of the model, however, may not fit every enterprise's needs or may not be feasible or practical to implement, especially when the enterprise initiates the philosophy of truly managing its information environment.

The implementation of the concept of managing the information environment should be evolutionary, and the IRDS aspects presented herein should undergo a rigorous cost/benefit analysis. This analysis should examine the start-up costs, because the IRM philosophy results in a front-end loading of costs. The benefits and reduced costs of information systems will accrue later, although some short-term benefits may occur, depending on the situation within the enterprise. The long-term benefits can accrue through standardization, improved communication, maintenance and modification of automated and manual information systems, better planning and better design of automated information systems, and databases which can provide more timely response to changes in the information environment, faster reaction to government legislation in the areas of privacy and paperwork controls, reducing the costs of conversion, and modeling of the information system environment to develop a more effective and efficient enterprise information environment. These benefits can accrue over time, but they are not free.

To realize the managed information environment will require organization changes, the development of user procedures for all information systems, the training of people, and the changing of people (both managers and workers) to a disciplined approach from the normal reactive approach. The degree of change to the enterprise is dependent on the present enterprise environment and the degree of enterprise adoption of the model presented.

Table 3-1

SLC PHASE: REQUIREMENTS

USES USERS	TECHNIQUES	METHODOLOGIES	NETWORKS	DATA FLOW	DATA STRUCTURE	INTEGRITY	VALIDATION CRITERIA	DATA DEFINITIONS	IMPACT ANALYSIS	SYSTEM DESIGN	BACKUP/ RECOVERY	CONSTRAINTS	MONITOR USE	CONVERSION	CHANGE MANAGEMENT	I/O FORMATS	ENCODE/ DECODE	GENERATE PROGS	GENERATE TEST DATA	VERSION CONTROL
INFO RESOURCE MGR		T		D					D											
SYSTEM DESIGNER																				
PROGRAMMER																				
CUSTOMER		D		I					I											
OPERATOR																				
PROJECT MANAGER																				
MAINTAINER																				
DATABASE ADMIN																				
SYSTEM ANALYST																				
AUDITOR		R																		
PROGRAMS																				

CODES: D (DEFINE), I (INFLUENCE), R (REVIEW), T (TRANSLATE), U (UPDATE)

Table 3-2

SLC PHASES: DEVELOPMENT/ANALYSIS

USES USERS	TECHNIQUES	METHODOLOGIES	NETWORKS	DATA FLOW	DATA STRUCTURE	INTEGRITY	VALIDATION CRITERIA	DATA DEFINITIONS	IMPACT ANALYSIS	SYSTEM DESIGN	BACKUP/ RECOVERY	CONSTRAINTS	MONITOR USE	CONVERSION	CHANGE MANAGEMENT	I/O FORMATS	ENCODE/ DECODE	GENERATE PROGRAMS	GENERATE TEST DATA	VERSION CONTROL
INFO RESOURCE MGR				R			R				R					R				
SYSTEM DESIGNER				U			U				U					U				
PROGRAMMER																				
CUSTOMER				R			D				I					D				
OPERATOR																				
PROJECT MANAGER				R			R				R					R				
MAINTAINER																				
DATABASE ADMIN				R			I													
SYSTEM ANALYST				I			I				I					R				
AUDITOR																				
PROGRAMS																				

CODES: D (DEFINE), I (INFLUENCE), R (REVIEW), T (TRANSLATE), U (UPDATE)

Table 3-3

SLC PHASE: DEVELOPMENT/DESIGN

USES / USERS	TECHNIQUES	METHODOLOGIES	NETWORKS	DATA FLOW	DATA STRUCTURE	INTEGRITY	VALIDATION CRITERIA	DATA DEFINITIONS	IMPACT ANALYSIS	SYSTEM DESIGN	BACKUP/ RECOVERY	CONSTRAINTS	MONITOR USE	CONVERSION	CHANGE MANAGEMENT	I/O FORMATS	ENCODE/ DECODE	GENERATE PROGRAMS	GENERATE TEST DATA	VERSION CONTROL
INFO RESOURCE MGR												D	D				D			
SYSTEM DESIGNER	I	D	D	R		D				D			I				I			
PROGRAMMER																				
CUSTOMER				I																
OPERATOR																				
PROJECT MANAGER	R	R	R	R		R	R			R	R	R	R			R	R			
MAINTAINER																				
DATABASE ADMIN	I		D				I			T	T	R	R			I	I			
SYSTEM ANALYST	D	I	R	R		R	R			T	R	R	I			R	I			
AUDITOR																				
PROGRAMS																				

CODES: D (DEFINE), I (INFLUENCE), R (REVIEW), T (TRANSLATE), U(UPDATE)

Table 3-4

SLC PHASE: DEVELOPMENT/PROGRAMMING

USES / USERS	TECHNIQUES	METHODOLOGIES	NETWORKS	DATA FLOW	DATA STRUCTURE	INTEGRITY	VALIDATION CRITERIA	DATA DEFINITIONS	IMPACT ANALYSIS	SYSTEM DESIGN	BACKUP/ RECOVERY	CONSTRAINTS	MONITOR USE	CONVERSION	CHANGE MANAGEMENT	I/O FORMATS	ENCODE/ DECODE	GENERATE PROGRAMS	GENERATE TEST DATA	VERSION CONTROL
INFO RESOURCE MGR																				
SYSTEM DESIGNER																				
PROGRAMMER	R	R		R	R	R	R	R	R					R			R	R	R	R
CUSTOMER																				
OPERATOR																				
PROJECT MANAGER	R	R		R	R	R	R	R	R					R			R	R	R	R
MAINTAINER																				
DATABASE ADMIN					D															
SYSTEM ANALYST																				
AUDITOR	R	R		R	R	R	R	R	R					R			R	R	R	R
PROGRAMS																				

CODES: D (DEFINE), I (INFLUENCE), R (REVIEW), T (TRANSLATE), U (UPDATE)

SLC PHASE: INSTALLATION

USERS \ USES	TECHNIQUES	METHODOLOGIES	NETWORKS	DATA FLOW	DATA STRUCTURE	INTEGRITY	VALIDATION CRITERIA	DATA DEFINITIONS	IMPACT ANALYSIS	SYSTEM DESIGN	BACKUP/ RECOVERY	CONSTRAINTS	MONITOR USE	CONVERSION	CHANGE MANAGEMENT	I/O FORMATS	ENCODE/ DECODE	GENERATE PROGRAMS	GENERATE TEST DATA	VERSION CONTROL
INFO RESOURCE MGR													R				U			
SYSTEM DESIGNER																				
PROGRAMMER																				
CUSTOMER				R			R				R		R						D	
OPERATOR	R	R									R									
PROJECT MANAGER	R	R		R			R				R		R	R			R		R	
MAINTAINER																				
DATABASE ADMIN											R		R							
SYSTEM ANALYST																			D	
AUDITOR	R	R		R			R				R		R	R			R		R	
PROGRAMS							R										R			

CODES: D (DEFINE), I (INFLUENCE), R (REVIEW), T (TRANSLATE), U (UPDATE)

Table 3-5

Table 3-6

SLC PHASE: OPERATION

USES USERS	TECHNIQUES	METHODOLOGIES	NETWORKS	DATA FLOW	DATA STRUCTURE	INTEGRITY	VALIDATION CRITERIA	DATA DEFINITIONS	IMPACT ANALYSIS	SYSTEM DESIGN	BACKUP/ RECOVERY	CONSTRAINTS	MONITOR USE	CONVERSION	CHANGE MANAGEMENT	I/O FORMATS	ENCODE/ DECODE	GENERATE PROGRAMS	GENERATE TEST DATA	VERSION CONTROL
INFO RESOURCE MGR						R					R	R	R							
SYSTEM DESIGNER																				
PROGRAMMER																				
CUSTOMER						R					R	R	R							
OPERATOR											R									R
PROJECT MANAGER																				
MAINTAINER																				
DATABASE ADMIN				R		R					R	R	R							D
SYSTEM ANALYST																				
AUDITOR	R			R	R	R	R				R	R	R				R			R
PROGRAMS					R												R			

CODES: D (DEFINE), I (INFLUENCE), R (REVIEW), T (TRANSLATE), U (UPDATE)

SLC PHASE: MAINTENANCE/MODIFICATION

Table 3-7

USES / USERS	TECHNIQUES	METHODOLOGIES	NETWORKS	DATA FLOW	DATA STRUCTURE	INTEGRITY	VALIDATION CRITERIA	DATA DEFINITIONS	IMPACT ANALYSIS	SYSTEM DESIGN	BACKUP/ RECOVERY	CONSTRAINTS	MONITOR USE	CONVERSION	CHANGE MANAGEMENT	I/O FORMATS	ENCODE/ DECODE	GENERATE PROGRAMS	GENERATE TEST DATA	VERSION CONTROL
INFO RESOURCE MGR		T		D		R			D		R	D,R	D,R				D,U			
SYSTEM DESIGNER	I	D	D	R		D				D			I				I			
PROGRAMMER	R	R		R	R	R	R	R	R					R			R	R	R	R
CUSTOMER		D		R,I		R	R		I		R	D,R	R				I		D	
OPERATOR	R	R									R									R
PROJECT MANAGER	R	R	R	R	R	R	R	R	R	R	R	R	R	R		R	R	R	R	R
MAINTAINER																				
DATABASE ADMIN	I		D	R		R	I			T	T,R	R	R			I	I			D
SYSTEM ANALYST	D	I	D	R		R	R			T	R	R	I			R	I			
AUDITOR	R	R		R	R	R	R	R	R		R	R	R	R			R	R	R	R
PROGRAMS					R		R										R			

CODES: D (DEFINE), I (INFLUENCE), R (REVIEW), T (TRANSLATE), U (UPDATE)

THIS TABLE IS AN OVERLAY OF TABLES 3-1 THROUGH 3-6;

THE APPLICABLE PORTION DEPENDS ON THE MODIFICATION/MAINTENANCE

FUNCTIONS USERS	ENTERPRISE-ORIENTED	DIRECTORY	SYSTEM SPECIFIC
AUDITORS	X	X	X
PLANNERS	X		
FUNCTIONAL USERS		X	X
ADP MANAGERS			X
ADP PROGRAMMERS			X
DATABASE ADMINISTRATORS			X
PROGRAMS			X
DATABASE DESIGNERS	X		X
DATA ADMINISTRATORS	X		

Table 3-8

USERS OF THE IRDS:
DATABASE PERSPECTIVE

FUNCTIONS USES	ENTERPRISE-ORIENTED	DIRECTORY	SYSTEM SPECIFIC
DESCRIBING DATA	X		
DESCRIBING DATABASES			X
RECORDING LOGICAL VIEWS	X		
MONITORING USE			X
ANALYZING USE	X		X
DEFINING INTEGRITY	X		
GENERATING SCHEMA AND SUBSCHEMA			X
IMPLEMENTING INTEGRITY			X
IMPACT ANALYSIS	X		
VALIDATION CRITERIA	X		X
ACCESS CONSTRAINTS			X
CROSS REFERENCE DATA USES/USERS	X		X
SYNONYMS	X		X
ENCODE/DECODE			X
SCHEMA VERSION	X		
DATABASE VERSION			X

Table 3-9

USES OF THE IRDS: DATABASE PERSPECTIVE

4. IRM POLICIES AND CONTROLS

Henry C. Lefkovits

CHAIRMAN

Biographical Sketch

Henry C. Lefkovits is President and Principal Consultant of Alpha Omega Group, Inc., a company that specializes in the solution of a broad range of problems in the area of Information Resource Management. He is the author of "Data Dictionary Systems," published by Q.E.D. Information Sciences, which presents an indepth analysis of commercially available data dictionary systems, and is currently working on the second edition of this work. He has been consulting in the field of data administration and the use of data dictionary systems, and is a frequent lecturer on these subjects. Dr. Lefkovits has been a member of the ANSI/X3/SPARC Database Systems Study Group since its inception. He chaired the Data Dictionary Systems Study Group which originated the SD-3 on Information Resource Dictionary Systems, and then acted as the Convener of X3H4, the Technical Committee on IRDS. Dr. Lefkovits holds B.A. and M.A. degrees from the University of Texas and a Ph.D. in mathematics from Rice University.

PARTICIPANTS

Susan Bidwell
Craig Cook
Tom Fitzgerald
George Gajnak
Scott Hightower

Henry Lefkovits
Belkis Leong-Hong
Mike Meyer
Richard Mixon

William Perry
Zella Ruthberg
Dennis Shaw
David Vincent
Ron Voell

4.1 INTRODUCTION

As its first agenda item, the panel on IRM Policies and Controls decided to agree on a working definition of Information Resource Management. Such a definition was required to set the boundaries of discourse for the work of the panel and to give the proper perspective to the integration of the wide-ranging technologies, users, and areas of expertise that converge in IRM.

4.2 DEFINITION OF IRM

> Information Resource Management is whatever policy, action, or procedure concerning information (both automated and non-automated) which management establishes to serve the overall current and future needs of the enterprise. Such policies, etc., would include considerations of availability, timeliness, accuracy, integrity, privacy, security, auditability, ownership, use, and cost-effectiveness.

The IRM definition emphasizes the enterprise-wide nature of planning and execution of information policies, actions, and procedures in order that data can be treated as a true resource of the enterprise. It also reflects the primary shift of EDP uses from processing-centered design methodologies to data-centered design methodologies.

This definition was again reviewed upon completion of the work of the panel and a general consensus existed as to its usefulness and proper focus on the problems that were discussed.

4.3 SYSTEM LIFE CYCLE

The scope of IRM and its impact extend into a large number of operations of an enterprise. Since the work of the panel was limited by the relatively short duration of the Workshop, the decision was made by the panel members to limit their considerations to the analysis of IRM impacts on the area of computer-based systems. In order to organize the work to be done and to properly highlight the applicability of IRM concepts throughout the life of such systems, it was decided to adopt a particular definition of a System Life Cycle (SLC) and to categorize IRM impacts according to its phases.

The following IRM phases were adopted:

- The Requirements Determination Phase consists of defining the user-visible functionality, including data elements required.
- The Data/System Specification Phase consists of translating the results of the previous phase into an implementable specification. This phase includes a logical database design. The phase should also address integration plans in the sense of what is done about existing systems and data that will be replaced or interfaced within the new system.
- The Database/System Design Phase consists of detailed program designs and specifications and physical database specifications that will achieve the functionality outlined in the previous two phases.
- The Construction/Testing Phase consists of program and database implementation coding and unit testing.
- The Integration/System Testing Phase results in a system ready to be placed in operation.
- The Installation/Operation Phase is the introduction and then routine production use of the system.
- The Maintenance Phase is routine repair of the system.
- The Enhancement Phase is the addition of new functionality resulting from use of the system.
- The Termination Phase is the archiving and termination of production uses of the system.

4.4 ORGANIZATION OF THE PANEL

The members of the panel had been drawn from both the technology sector and the internal auditing profession. It was decided that the value of the work would be enhanced if the panel spent some of its working time divided into separate technology and auditing groups charged with different but complementary missions.

The technology group was charged with identifying issues associated with computer-based tools which are applicable and available to support individual SLC phases from a developer's point of view.

The emphasis of the auditing group was to explore the exposures that existed in the various SLC phases and to suggest tools and procedures that would minimize the risk of these exposures.

Both groups were responsible for developing lists of IRM policies, required deliverables, and concerns for the various SLC phases.

In the final working session of the panel, each group presented its findings and a comparison was made. Since the limited time available to each group did not allow exhaustive lists to be prepared, it was not expected that a complete overlap between the respective findings would exist. The analysis did however reveal a substantial overlap in procedures that IRM would develop and exposures that were identified. As such, the general conclusion was that the development of IRM procedures would significantly reduce the risk exposures that were present in the SLC phases.

In the following, the findings of the panel are presented. It must be pointed out that the use of Dictionary Systems as a tool is not highlighted as it was felt that this would occur in the work of other workshop panels. Instead, the findings were structured to focus more on IRM-related issues.

4.5 IRM POLICY RECOMMENDATIONS

The panel developed a set of IRM policies for a hypothetical enterprise. These results are grouped in the following manner:

- Fundamental IRM policies (Sections 4.6 - 4.7)
- IRM policies applicable across SLC phases (Section 4.8)
- IRM policies applicable to individual SLC phases (Section 4.9)

The results presented are intended to be representative; in no instance does there exist the implication that the results are exhaustive.

4.6 FUNDAMENTAL IRM POLICIES

The panel proposes the following fundamental IRM policies:

* There will be a Strategic System Architecture Plan for the enterprise.

* There will be a Long Range Information Plan which considers short- and long-range cost aspects of maintaining enterprise data.

* There will be Guidelines for the use of database management systems, languages, networks, dictionary systems, hardware, and other system components.

* There will be a Data Standardization Program.

4.7 STRATEGIC SYSTEM ARCHITECTURE AND INFORMATION PLANS

The Strategic System Architecture and Information Plans constitute the nucleus of the documents called for by the Fundamental IRM Policies. The panel did not discuss the specific contents of these documents; however, it is felt that the present report will be more meaningful if a more detailed description of them were included. What follows is an outline and brief description of each chapter which can be considered representative of the nature of these documents:

Table 4-1

STRATEGIC SYSTEM ARCHITECTURE PLAN OUTLINE

CH	TITLE	PURPOSE OF CHAPTER
1	Intro.	1. Management overview.
2	The Organization	1. Description of organization in terms of high-level enterprise event model (normative). 2. Diagram of high-level data resource model (normative). 3. Business objectives and goals.
3	IRM Goals	1. Obtain agreement on goals and principles which are not dependent on hardware, software, or application package commitments.
4	IRM Initiatives	1. Describe the environmental IRM initiatives. 2. Describe enterprise event support initiatives (application areas).
5	Data Resource Model	1. Diagram and describe the conceptual data model for the enterprise. 2. Outline plan for model refinement and translation to target database design and physical database/ file schemas.
6	Enterprise Event Model	1. Diagram the current event model and describe the planning, control, and operational events for the enterprise. 2. Outline plans for model refinement and collection of cost/benefit data for each event.
-	Attachments	1. Description of current EDP systems in operation and development. 2. Lists of current hardware installed. 3. The technology forecast for predicting impacts on the enterprise for the next 5 years. 4. Glossary of terms used. 5. Bibliography and references used.

There are several purposes in modeling the organization around enterprise event processes:

- The enterprise event modeling focuses cost, productivity improvements, and other organizational benefits on specific business practices (i.e. the events).
- The event modeling also facilitates organizational development processes to occur in a neutral discussion environment which is not tied to existing departments and applications.
- The event modeling also facilitates planned migration of current large, batch processing systems into terminal-oriented on-line systems (where appropriate). Each of the new support applications will be self-contained from a development viewpoint, but feeding from and to the central database (where appropriate).
- The Long-Range Information Plan would be a collection of detailed project plans (objectives, milestones, and resource estimates) to carry out the Architecture Plan. The Information Plan could be included as part of the Architecture Plan itself or as a separate document, depending on the planning and project control styles and policies of the enterprise.

4.8 IRM POLICIES APPLICABLE ACROSS ALL SLC PHASES

The panel discovered that there are a small group of policies which are not fundamental to the overall success of IRM, but nevertheless apply across all SLC phases. These IRM policies exist in the following areas:

- Project Management
- Configuration Management
- Metadata Management
- Documentation Management
- Review Management

For each of these policy areas subsequent sections discuss or list deliverables, applicable computer-based tools, concerns about tools, and recommended IRM control policies.

4.8.1 Project Management.

The management policies and tools for projects should be integrated with the other aspects of IRM.

a. Deliverables:

- Estimation aids
- Planning aids
- Measurement aids
- Project control aids
- Scheduling aids
- Organizational development aids

b. Tools:

- Automated project management and estimating tools (e.g. PC70, PAC-II, etc.)
- Data dictionary system

c. Concerns about Tools:

- Project management data must integrate into enterprise dictionary system.

d. IRM Control Policies:

- IRM requires an enterprise policy regarding specific project management data and processes.

4.8.2 Configuration Management.

Configuration Management (CM) applies software engineering principles to pinpoint responsibilities for maintaining software and hardware baseline inventories and all changes to the baseline. CM requires both policy direction and integrated tool support.

a. Deliverables:

- Change control aids
- Software/hardware baseline inventory aids
- Status accounting which fixes individual responsibility for actions and states
- Pointer system to baseline references and documents not contained within the CM system

b. Tools:

- Status accounting packages (ANMP)
- Data dictionary systems
- Source and object library packages (e.g. Librarian, Panvalet, etc.)

c. Concerns about Tools:

- Configuration management data must integrate into enterprise dictionary system.

d. IRM Control Policies:

- Any change to baseline requires passing through change control procedures.

4.8.3 Metadata Management.

The use of computer data dictionary systems is expected to be the main integrating tool of IRM. This dictates that IRM policies and procedures be set up to manage the structure and data content of the dictionary itself.

a. Deliverables:

- Data definitions
- Process definitions
- Environment definitions

b. Tools:

- Existing, first-generation dictionary systems (e.g. Data Manager, UCC10, etc.)
- Proposed second-generation dictionary systems
- Modeling tools (e.g. PSL/PSA, etc.)

c. Concerns about Tools:

- Automated tools must be friendly to users.
- Metadata management must not disrupt the on-going integration support of the enterprise dictionary system.
- Multiple sites/systems must be addressed.
- Security of dictionary systems needs to be strong.

d. IRM Control Policies:

- All metadata is collected and documented according to enterprise standards.

4.8.4 Documentation Management.

Software and system documentation is a critical aspect of maintaining a viable IRM environment. IRM tools and policies are needed to manage the documentation creation, maintenance, and retrieval processes.

a. Deliverables:

- Development and maintenance standards on-line (e.g. forms, methodologies, examples, etc.)
- Word-processing input/edit features to make creation and maintenance of documents easy and controlled for all users (authors, reviewers, and readers)
- Documents (manuals, specifications, etc.)
- References to documents not contained within the system itself

b. Tools:

- Word processors
- Text editors
- Language conversion aids (e.g. compilers, interpreters, etc.)
- Syntax checkers for languages
- Standards checkers for languages and documents
- System analysis/design methodologies
- Graphic aids
- Electronic mail and teleconferencing aids

c. Concerns about Tools:

- Textual data creation, maintenance, and retrieval must be integrated by or through the enterprise dictionary system.
- Where do the language conversion aids and syntax checkers belong?
- Where does configuration management stop and documentation management begin?

d. IRM Control Policies:

- There is a baseline specification of what constitutes documentation (i.e. most approved deliverables from each SLC phase).

4.8.5 Review Management.

Formal review groups will become the governing authority of IRM Life Cycle Management. The convening, scheduling, and reporting of the work of these review groups require IRM policies and tools in order to function effectively.

a. Deliverables:

- Review schedules (participants, time, place, agendas, etc.)
- Approval certification of baseline documentation, software, and hardware
- Review and exception reports
- "GO"/"NO-GO" decisions on key milestones

b. Tools:

- Audit packages (operational systems)
- Verification and validation aids (development systems)
- Standards compliance checkers

c. Concerns about Tools:

- Review management data must integrate into enterprise dictionary system.
- SLC review management tools are needed.
- Better audit tools are needed to cover a broader scope (e.g. privacy, security, cost, data independence, redundancy, conversion, etc.).

d. IRM Control Policies:

- No system may change state (i.e. proceed to another SLC phase) without a "GO" decision properly documented under the Review Management procedures.

4.9 IRM POLICIES FOR SPECIFIC SLC PHASES

The same categories are given below for policies and tools applicable to specific SLC phases. In addition, lists of concerns and exposures for auditing individual activities within phases are given. The phases described are:

* Requirements Determination Phase

- * Data/System Specifications Phase
- * Database/System Design Phase
- * Construction/Testing Phase
- * Integration/System Testing Phase
- * Installation/Operation Phase
- * Maintenance Phase
- * Enhancement and Termination Phases

4.9.1 Requirements Determination Phase.

This phase initiates a new SLC project by systematically documenting and obtaining approval of user requirements for the project.

a. Deliverables (Data and System Requirements):

- Data/data structures
- User-visible function statements (including user manual and user-visible error codes)
- Environmental statements and constraints

b. Tools:

- Dictionary system features to compare strategic plan statements with data and system requirements statements
- Automated requirements tools
- Graphic entity-relationship diagram aids
- Dataflow diagram aids
- Data event/data task model aids
- Structured analysis aids (e.g., BSP, SADT)
- Organization chart aids
- Transaction matrix aids

c. Concerns about Tools:

- Requirements data must integrate into enterprise dictionary system.
- Better requirements tools are needed.

d. IRM Control Policies:

- System requirements will be consistent with the strategic system architecture plan.
- System requirements will be in standard form.
- Enterprise security, privacy, and retention policies will be followed in setting requirements.

e. Auditing Concerns and Exposures:

1. Preliminary to any System Life Cycle:

- Standards are in place and well documented for:
    - programming structures
    - systems development
    - programming languages
    - cost/benefit analysis
    - feasibility studies
- Organizational segregation of functions is satisfactory.
- Training is planned and accomplished.

2. User Visible Functions:

- Problem definition
- Inconsistency
- Standards adherence
- Procedural adherence
- Volumes
- Clarity
- Completeness
- Legitimacy
- Reasonableness

3. Data Definition (Applies to Entities, Relationships, and Attributes):

- Incompleteness of definition
- Omissions
- Extraneous
- Redundant
- Inconsistency
- Units/value sets
- Sensitivity
- Responsibility for data and access retention volumes

4. Environmental:

- Availability
- Volumes
- Response time
- Terminal characteristics
- Security
- Communications characteristics
- Physical environment
- Personnel (skills)

4.9.2 Data/System Specification Phase.

This phase establishes the overall architecture for the project. Subsequent detailed design and construction phases can be accomplished independently over time and place, in most cases.

a. Deliverables (Logical Design):

- Logical database/file design
- Logical system design (including transactions, forms, reports, screens)
- Interface specifications (integration with other systems)
- Alternatives considered (feasibility)
- Test plan
- Cost/benefit analysis
- Hardware/capacity requirements

b. Tools:

- Database/file design aids
- Screen/report design aids
- Hardware configuration aids
- Capacity modeling aids
- Simulators/application prototyping aids

c. Concerns about Tools:

- Design data must integrate into enterprise dictionary system.
- Better design tools are needed.

d. IRM Control Policies:

- Data designs will support the enterprise conceptual data model.
- Design specifications will be in standard form (e.g., dictionary).

- Forms and reports (input/output) will be controlled as part of configuration management.

e. Auditing Concerns and Exposures:

1. Interface To Other Systems:

   - Redundancy
   - Compatibility
   - Security
   - Privacy
   - Completeness
   - Impact of change

2. System Design Legitimacy of Transactions

3. Conceptual Data Structure Design:

   - Existence
   - Reflection/adherence in logical design

4. Preliminary Cost/Benefit Analysis:

   - Approval levels
   - Reasonableness
   - Segmentation

5. Phasing Strategy:

   - Feasibility
   - Reasonableness
   - Legality
   - Resource availability
   - Training
   - Timeliness
   - Manageability

6. Conversion of Software/Data:

   - Existing and target data
   - Existing and target software
   - Existing and target systems
   - Existing and target hardware

4.9.3 Database/System Design Phase.

This phase completes the detailed design of both database and processing modules necessary before construction begins. The designs must not violate the architecture laid down in the previous phase.

a. Deliverables (Detailed Specifications):

- Physical database/file design
- System specifications (including inputs, outputs, networks, and hardware)
- Test specifications
- Revised user documentation
- "Hard" cost/benefit analysis
- Conversion specifications
- Training plan

b. Tools:

- Security software (e.g. RACF, ACF2)
- File and database optimization packages
- Conversion aids
- Test data generators
- Training aids
- Structured system design aids

c. Concerns about Tools:

- Specification data must be integrated into enterprise dictionary system.
- Better specification tools are needed.

d. IRM Control Policies:

- Data specifications must be validated to show that they support logical design.
- System specifications must be validated to show that they support the logical system design.
- Data specifications must be validated to support the data standardization program, including sources and ownership of all data.
- Design data will be in standard form (e.g. dictionary).

e. Auditing Concerns and Exposures:

1. System Specification:

- Legitimacy
- Ownership/access approval
- Synchronization--distributed data processing
- Data dictionary system use in operational system
- File sensitivity
- I/O specs
- Test plan
- Conversion plan
- Backup/recovery/restart/reorganization plans
- System maintenance controls (i.e. run to run)

2. Schemas/Sub-schemas:

- Ownership/access approval
- Integrity
- Data dictionary system interfaces
- Database administration
- User training

3. Data Communications Risks:

- Backup/restart/recovery handling
- Adequacy of data communications
- Encryption requirements
- Protocols
- Networking configuration and review

4. Finalized Cost/Benefit Analysis:

- Approval levels are correct and completed.
- Reasonableness
- Segmentation

5. Security/Privacy:

- Access restrictions (Password/access matrix administration)
- Error reporting
- Sensitivity levels
- Penetration reporting
- Data security--strategy trade-offs
- Physical security requirements (e.g. terminals, files)
- External regulations--compliance

6. Documentation Risks:

- Draft user manual
- Start of operations manual

7. Global Review:

- Cross check to requirements.

4.9.4 Construction/Testing Phase.

This phase is made less labor intensive through the work of the preceding planning and design phases and through the use of an integrated IRM tool support kit. The unit testing in this phase validates the work of individual programmers.

a. Deliverables:

- Coding (programs, job control statements, maps, object code)
- Program documentation
- Test data and results
- Training materials

b. Tools:

- Test data generators
- Source language generators
- Compilers and interpreters
- Documentation aids
- File/database utilities

c. Concerns about Tools:

- Construction/testing data must be integrated into enterprise dictionary system.

d. IRM Control Policies:

- Programs will comply with the enterprise data standardization project, including dictionary system-generated data definition sections.
- Coding must be validated back to the detailed specifications.

e. Auditing Concerns and Exposures:

1. Program Logic:

- Adequacy
- Adherence to user requirements
- Specialized audit requirements (e.g. SCARF, tagging, ITF)

2. Program Coding:

- Adherence to standards for coding
- Existence of structured reviews/walkthroughs
- Audit participation for critical functions in structured reviews/walkthroughs
- Existence and quality of imbedded documentation
- Data omissions discovered in programming
- Resulting data dictionary system update/modifications
- Synchronization of dictionary to environment
- Security of libraries (source, load)
- Resolution of design/logic errors

3. Unit/String Testing:

- Existence
- Construct systems test base.

4. Documentation:

- Final user's manual
- Final operations manual
- Training program

4.9.5 Integration/System Testing Phase.

The testing in this phase proves that the software and database modules constructed independently under an overall design structure will operate together in a pseudo-production environment.

a. Deliverables:

- Scheduling aids
- Recovery and restart procedures
- Training accomplished
- Completed system (baselined)

b. Tools:

- Recovery and restart aids
- Backup and restoration aids
- Training aids

- Configuration management aids

c. Concerns about Tools:

- Testing and validation data must be integrated into enterprise dictionary system.

d. IRM Control Policies:

- The integration testing plan and execution must be accomplished independently from unit tests.

e. Auditing Concerns and Exposures:

- User involvement/independent test
- Restart/recovery test--all conditions
- Stress test/stump the system
- Volume test
- Finalization of data conversion plans
- Proof and balancing controls
- Acceptance test by user
- Security violation testing
- Validation of production job control language
- Production environment creation (i.e. library)
- Parallel test plans (if required)

4.9.6 Installation/Operations Phase.

This phase installs the new modules and verifies that they can operate successfully in a production environment.

a. Deliverables:

- Performance data
- Post-installation audit results
- Trouble logs
- Running system
- Attempted security violations documented
- Production files/database
- Production job control language
- Production program libraries

b. Tools:

- Performance measurement aids
- Security packages
- DBMS statistics aids
- Dictionary system (end user interface, problem tracking, scheduling, security violations)

- Restart/recovery aids
- Backup/restoration aids

c. Concerns about Tools:

- Operations data, including scheduling, must integrate into enterprise dictionary system.

d. IRM Control Policies:

- Security policy will be audited and enforced.
- Performance data will be used as a feedback mechanism for tuning, configuration changes, and long-range capacity planning.

e. Auditing Concerns and Exposures:

1. Installation Review:

- Documentation of emergency modifications
- Program change control

2. Post-installation Audit (6 months later):

- Verify cost savings (operations, development).
- Verify changes to critical modules.
- Insure auditability.
- Insure feasibility of restart/recovery and alternate site production.
- Analyze change control log.
- Examine implementation of user and operations training.

4.9.7 Maintenance Phase.

The amount of effort required to maintain a system can be significantly reduced if preceding phases have followed IRM policies consistently using a well-integrated set of IRM tools. Most of the IRM tools can also be used during the maintenance phase.

a. Deliverables:

- Changes necessary to meet baseline requirements

b. Tools:

- Security packages
- Configuration management packages

- Database integrity checkers

c. Concerns about Tools:

- Maintenance data and changes must integrate into enterprise dictionary system.

d. IRM Control Policies:

- All changes controlled through configuration management (dictionary system) facilities
- Separation of production and test environment through dictionary system facilities

e. Auditing Concerns and Exposures:

- Compliance with policies on user initiated/approved changes
- Identification of major enhancements
- Hardware/system software configuration control
- Performance monitoring function

4.9.8 Enhancement and Termination Phases.

The panel considers that the enhancement phase consists of restarting the SLC cycle at an appropriate phase and using the same tools and policies as for a development project.

The termination phase consists of suitable archive procedures and aids so that systems that are no longer operational can be resurrected, on demand.

4.10 CONCLUSION AND SUMMARY

An IRM definition is proposed that emphasizes the use of data as a true enterprise resource. IRM also emphasizes the use of data-centered design methodologies in place of earlier process-centered design techniques.

Fundamental IRM policies are proposed for a hypothetical organization. These policies emphasize the need for an integrated set of planning documents:

* Strategic System Architecture Plan

* Long Range Information Plan

88

* Guidelines for use of dictionary system, languages, database management systems, and networks

* Data Standardization Plan

A substantial impact of these IRM policies on all phases of the system life cycle of automated information systems can be expected. This impact will result not only in better cost effectiveness, but also in systems which are more satisfactory from an auditing point of view. It is to be expected that similar impacts will be found when similar principles are applied to other areas of the enterprise also involved with information resources.

5. LOGICAL DATABASE DESIGN

Shamkant B. Navathe

CHAIRMAN

Biographical Sketch

Sham Navathe is an Associate Professor of Computer and Information Science at the University of Florida, Gainesville. After receiving a Ph.D. from the University of Michigan he was on the faculty of the New York University Graduate School of Business Administration during 1975-79. He was an organizer of the NYU Symposium on Database Design in May, 1978 which was the first major meeting of database design experts from industry and academia. He has worked in the data processing industry with IBM and Electronic Data Systems. He has been affiliated with the Research Divisions of IBM in the U.S., and Siemens and GMD in West Germany.

Dr. Navathe′s research contributions are in the areas of data translation and restructuring, modeling and analysis of data, database design, and mappings in a multi-level DBMS architecture. He has worked on a cost-benefit model for data management systems, statistical database modeling, and office information systems.

PARTICIPANTS

Don Batory
Michael Brodie
Peter Buneman
Eric Clemons
Thomas Cook
Gary Craig
Robert Curtice
Paul Fehder
James Fry
Richard Godlove
Thomas Hester
Barry Housel
David Jefferson
Beverly Kahn
Larry Kerschberg
Dennis McLeod
Bruce Rollier
Richard Secrist
Jay Louise Weldon
Gio Wiederhold

5.1 INTRODUCTION

This panel addressed itself to the issues of information resource management as they relate to an understanding of the users' information requirements, techniques for representing and integrating the individual information models, and finally the overlap between logical and physical design. In keeping with the theme of this workshop, a constant reminder was given to the discussants to examine the role data dictionaries should play in assisting the process of logical database design.

The overall conclusions reached by the panel can be summarized as follows:

i) Logical database design is a complex activity. Computer-assisted methodologies will allow designers to cope with this problem more effectively when dealing with the information resources of a large organization. Use of semantic data models needs to be heavily stressed.

ii) Requirement analysis is an extremely important, initial phase of logical database design. Existing tools and methodologies have a number of desirable features, but there is no single technique which stands out.

iii) Existing schema design techniques in the form of commercial software products or prototype research tools are either grossly inadequate or not usable at all for realistic situations. The "tool workbench" concept was advanced as a means for integrating these tools.

iv) Data dictionary systems (DDSs) will have a lot to offer in the future for better database design. A group looked into the roles of DDSs in the management of data as a whole and concluded that DDSs and DBMSs deserve equal emphasis as tools for information resource management. Another group identified a list of desirable features for future DDSs to help in the design of databases for centralized as well as distributed environment.

v) A number of recommendations were made to deal with distributed databases and the allied problems of communication, control, etc.

In order to set a framework for discussion, the overall design process is described in Figure 5-1. Figure 5-2 describes the requirements analysis activity which feeds into logical design.

91

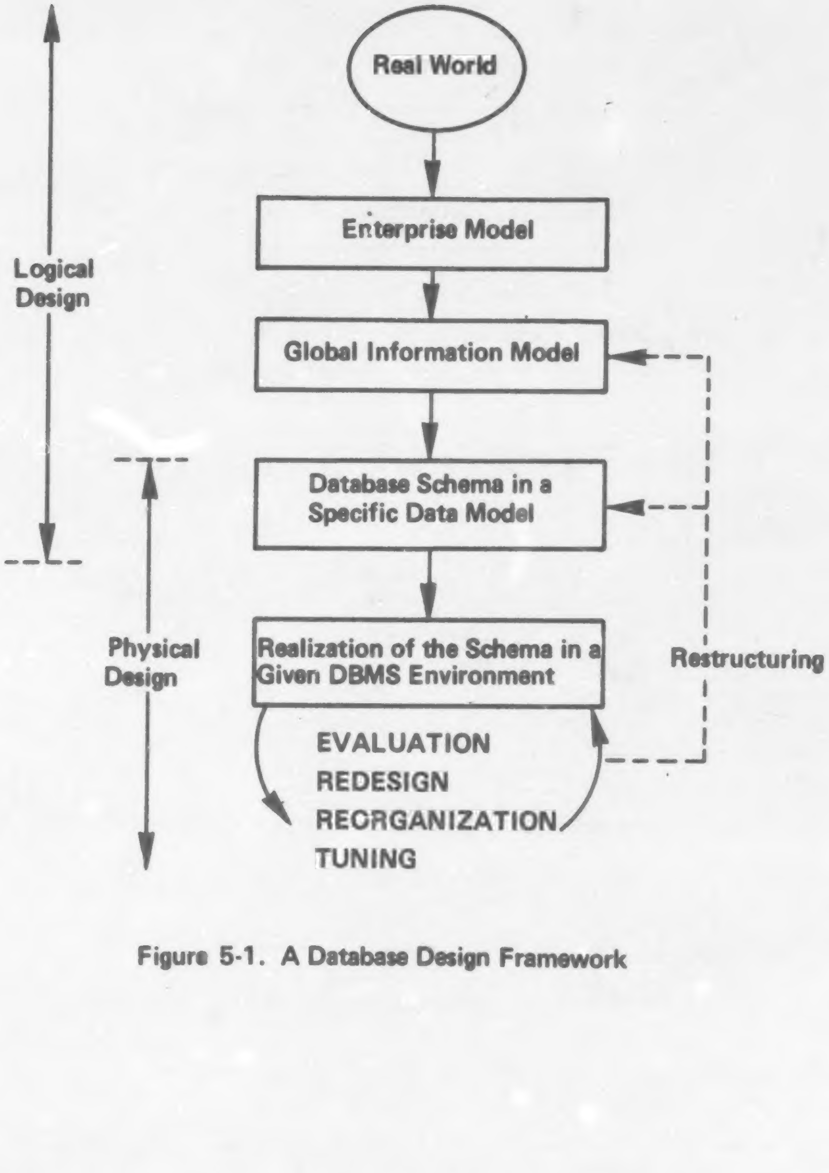
Real World
Enterprise Model
Logical Design
Global Information Model
Database Schema in a Specific Data Model
Physical Design
Realization of the Schema in a Given DBMS Environment
Restructuring
EVALUATION
REDESIGN
REORGANIZATION
TUNING

Figure 5-1. A Database Design Framework

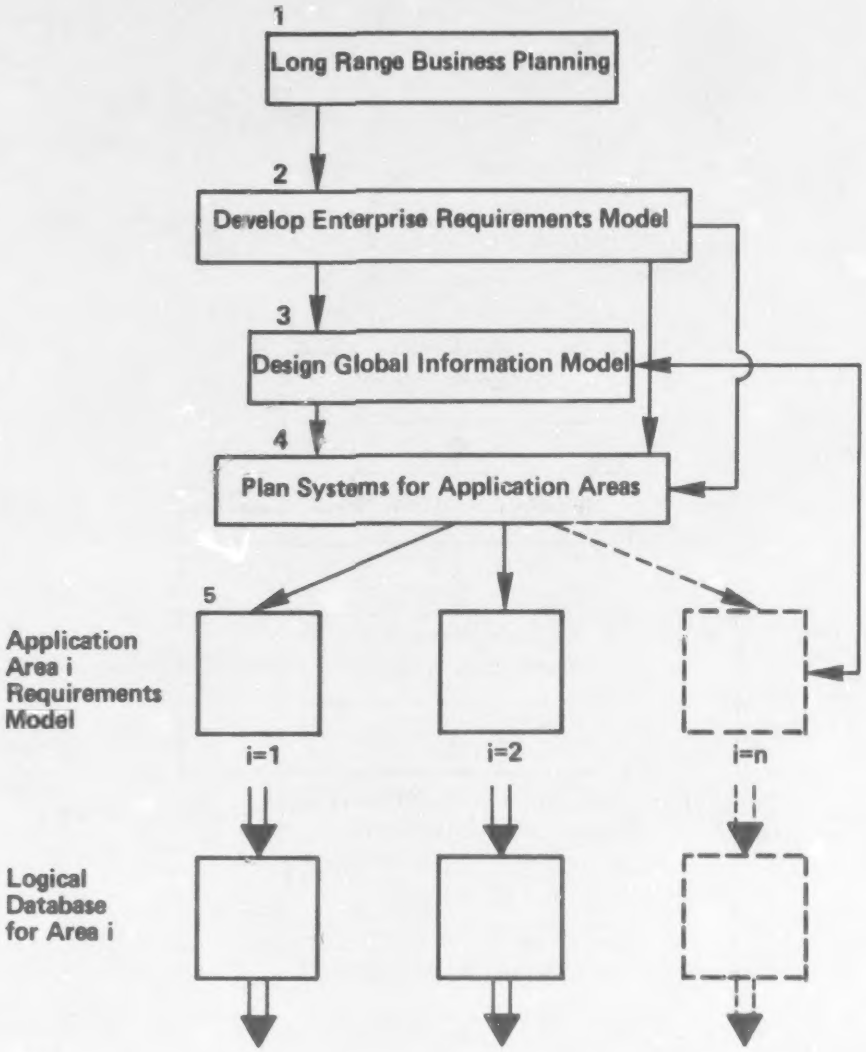
1
Long Range Business Planning
2
Develop Enterprise Requirements Model
3
Design Global Information Model
4
Plan Systems for Application Areas
5
Application
Area i
Requirements
Model
i=1
i=2
i=n
Logical
Database
for Area i

Figure 5-2. Requirements Analysis and Planning in an Information Resource Management Environment

Figure 5-1 shows the various models of information/data that exist in going from a universe of discourse drawn from the real world to a database which is populated with data stored on a physical machine-readable medium and is made available to users.

Figure 5-2 shows the five phases of requirements analysis and planning. Single arrows show sequences of action while double arrows show mappings among models.

The Enterprise Model comprises data expressed in terms of objects, things, activities, events, policies, concepts, etc., which refer to an overall enterprise. It also incorporates at a broad level the attributes and relationships among these objects. As an example, let us consider the university as an enterprise and suppose that the universe of discourse for a given context includes the instructors, the students, the physical facilities, including buildings and equipment, the courses, etc. In this case, the enterprise schema would include all the above objects plus the relevant associations:

e.g., a student enrolls in a course;
a course is taught by an instructor in a certain room;

each piece of equipment has a fixed maintenance schedule;

instructors teach labs which are special classes meeting in rooms designated as a lab, etc.

The above are a few of the possible relationships about which there may be a need to store data in the database. The Enterprise Model sometimes includes the type of processing to which the corresponding data is subjected. The Enterprise Model should contain relatively static or invariant data which characterizes the enterprise in terms of its relevant aspects for a particular database.

The Application Requirement Models shown in Figure 5-2 capture the dynamics of the enterprise-wide information by developing requirement specifications of individual applications. In the case of the above example of a university, the application requirement models may refer to applications such as registration, generation of class rosters, faculty office assignments, classroom assignments, preparation of inventories of equipment by buildings, departments or rooms, etc. In a top-down design methodology using successive refinements of specification, the applications are described at successively detailed levels.

The term Global Information Model refers to an output of the integration process which integrates the Enterprise Model with the individual application models. A global information model should typically be described by using a semantic model or a data abstraction model as a tool to capture the structure and semantics of relevant information as perceived by the entire user population as a whole. In its ideal form, the global information model idea is rather utopian. Organizations where user groups place widely varying demands on the data and, in fact, want to see their "own versions" of a database which are not necessarily compatible, may have to settle for more than one global information model.

More and more organizations are distributing the control of their data into the hands of local user groups. For them, the view integration activity proceeds first at the local level and then, depending on the need, a global information model may or may not be constructed. It has not been shown conclusively that a global information model is a must for distributed data management.

As shown in Figure 5-1, the next phase of design starts off with the global information model and ends with the realization of the schema in a specific DBMS environment, i.e. definition of the DDL, the integrity constraints, the privacy and security constraints and the processing of the database in that DBMS. For simplicity we shall refer to this phase as the schema design (SD) phase. It is obvious that the following overlaps exist:

a) SD phase overlaps with the process of view integration which is responsible for generating the global information model.

b) SD phase overlaps with the physical mapping of a DBMS schema into the corresponding files and/or storage structures in the DBMS.

It is difficult to draw well defined boundaries between physical and logical database design and further between SD and logical design or between SD and physical design.

5.1.1 Organization of the Panel

To accomplish the best results out of the two days of discussions among the panel members at the workshop, the panel was divided into the following groups. The results from each group are kept within one section of this chapter as far as possible.

Group I: Requirement Analysis/assessment of user needs.
GROUP LEADER: Weldon. MEMBERS: Curtice, Jefferson, Kahn.
Section 5.2 summarizes results of their discussion.

Group II: Information Modeling.
GROUP LEADER: Kerschberg. MEMBERS: Brodie, Buneman, Housel, McLeod, Wiederhold.
Section 5.3 summarizes their discussions.

Group III: Interface between Logical and Physical Design.
Don Batory.
Section 5.4 points out the basic issues of this interface.

Group IV: Role of Data Dictionary Systems in Management of Data and Database Design.
GROUP LEADER: Godlove. MEMBERS: Cook, Craig, Fehder, Hester, Rollier, Secrist.
Section 5.5 summarizes the overall thrust of their deliberations.

Each group leader submitted a report to Navathe after receiving their contributions from group members. Fry and Clemons participated only in the discussions. McLeod participated only in the writing. This chapter is an attempt at integrating these reports.

5.2 REQUIREMENTS ANALYSIS

5.2.0 Introduction

Database design is a process which is dependent on a good statement of the enterprise's (organization's) requirements. In this section of the report, we are concerned with requirements which affect the logical database design effort; moreover, we address this issue within the context of an Information Resource Management (IRM) philosophy.

In order to proceed further, we need to characterize more specifically what is meant by Information Resource Management. The main aspects of this approach are as follows:

- Information is to be regarded as a resource to the organization, requiring management on a broad level.
- The information resources should be shared across application and organizational divisions.
- Information consistency, privacy and security are of major importance.
- Data Management is to be independent of specific computerized application systems.

- Computerized application systems must be integrated. A major function within an independent business unit will constitute one application system (for example, the personnel function within a subsidiary company but not the materials management function within one factory).

The concept of Information Resource Management is still evolving, and the above characterizations need to be reviewed with this in mind. However, the essence of IRM is captured in the above points. Two very critical implications relating to requirements definition follow immediately from the above characterization. These are:

a) Under Information Resource Management, the requirements definition activities must be undertaken within the context of a top-down business planning process which has a broad scope, and relates information to business objectives. Without such a plan, there are no objectives for which to manage the information resources; IRM is undefined where business plans are absent. Currently, information is most readily related to tactical business objectives, although in the future it should be directed at strategic ones as well. To the degree that an IRM approach might fail, it is by far due to a lack of appreciation for this point.

b) While there are numerous tasks and operations which can and should be distributed in a modern environment, under Information Resource Management the following aspects of the requirements phase must be centralized:

- Coordination of requirements activities,
- Access to requirements documentation,
- Specification of the requirements definition methodology,
- Review and approval of requirements results, and
- Adjudication of disputes.

Note that this centralization is with respect to the scope of the IRM approach. Thus, it is possible to apply IRM to less than an entire organization. However, as pointed out above, there is some minimum scope to which IRM can reasonably be applied. For example, sharing data between two computer programs is not IRM.

5.2.1 The Activities of the Requirements Phase

In order to better understand the role of the requirements phase, we need to describe its immediate constituent steps and characterize the activities which precede and follow it in an idealized Information Resource Management environment. Figure 5-2 depicts the activities within and interfacing with requirements definition. We discuss each activity in turn.

5.2.1.1 Construct Long-Range Business Plan

This activity, while not strictly part of the Requirements Phase, provides critical inputs to it. These inputs define the nature of the organizational mission (the businesses it is in), the prioritized long-range objectives of the organization, and how information fits into these objectives. Additionally, it identifies any assumptions and constraints which are important to the organization's business. For example, we might assume that the Federal Trade Commission will not relax its consumer protection regulations concerning mail order businesses. Thus the business plan provides inputs in the forms of scope, objectives, assumptions, and constraints to the requirements process.

5.2.1.2 Develop Enterprise Requirements Model

This is the first task in the requirements phase. Its purpose is to provide a high level description of the objects, relationships, and functions within the Enterprise (organization). This description serves as a top-down constraint on more detailed requirements descriptions. It is usually done at the outset of IRM adoption, working from inputs provided by the business planning activity. Since by its very nature it is concerned with aspects very fundamental to the organization, it should be invariant over long periods of time. However, it may change as the result of merging two areas not previously integrated, (the result of a merger, for instance).

The purpose of the Enterprise Requirements Model is to provide:

a) The set of named object types of fundamental interest to the Enterprise.

b) A definition of each such object type including subtypes (i.e., the membership criteria for inclusion in the type).

c) Relationships (named only when appropriate) of fundamental interest among the object types.

d) Definition of high-level business functions and the object types and relationships involved in each.

For example, ACME Hotels might require PERSONS, with subtypes CUSTOMERS and EMPLOYEES, and ROOMS as fundamental objects of interest. CUSTOMERS may be involved in several relationships with ROOMS, including RESERVATION and OCCUPANCY. The fundamental business function of RENTING involves these objects and relationships.

The Enterprise model focuses on fundamental objects, relationships and functions of global interest. It is important that it ignore details of temporary or local interest only. It consists of three activities: data collection, specification, and analysis. Methods, tools, and techniques to accomplish these activities are discussed in the section 5.2.2.

5.2.1.3 Design (Initial) Global Information Model

The Global Information Model flows initially from the Enterprise Requirements Model. However, it can be expanded in scope and detail as more detailed requirements evolve from specific application areas. The Global Information Model differs from the other model in that it is concerned with the representation of objects and relationships as computerized data (which the Enterprise Model is not). Object identification schemes, the nature of the mapping between objects and their identifiers, domain integrity constraints and so on are in the purview of the Global Information Model. Its purpose is to provide top-down constraints on all further activities.

5.2.1.4 Plan Application Area Systems

This periodic planning activity determines the application systems to be implemented within the various application areas falling under the scope of the Enterprise. It sets the priority and sequence of these system development efforts, and is of interest here because it sets off the requirements effort associated with each area.

5.2.1.5 Develop Requirements Models for Application Areas

This effort consists of the same three activities (data collection, specification, and analysis) as the Enterprise requirements effort, but the objectives are somewhat different. The purpose of the Application Area requirements model is to provide:

- the set of all named object types of interest to an application area, including derivable ones, consistent with the Enterprise Requirements Model.

- definitions for these object types (as before), plus their computerized representation (consistent with or extending the Global Information Model), plus an approximation of the number of instances of each object type.

- all relationships among those object types of interest to an application area, consistent with the Enterprise Model, plus the nature of the mapping involved (one-to-many, mandatory, etc.).

- information processing requirements of the application area, including such characteristics as output data requirements, levels of summarization required, data and processing interactions, processing frequencies and cycles, and the like.

An application area is quite broad and may involve a fairly complex set of integrated systems. These systems may be implemented in a phased approach over a long time period. The interest of the application area requirements model is to provide data to the logical database design activity. A logical database design covering the application area is needed to support the integrated systems to be developed.

The requirements formulation should be both "bottom-up" (from examination of detailed user needs using techniques to be described below) and "top-down" (from constraints imposed by the Enterprise Requirements and Global Information Models). The mixture of top-down and bottom-up approaches is critical. The development of a complete top-down design for the entire organization has proven to be too time consuming and difficult -- management typically will not invest in such a large undertaking with a limited immediate value. On the other hand, Information Resource Management necessarily implies that an enterprise level view be developed and used to insure the sharing and consistency of data. We are convinced this cannot emerge as a result of "merging" or "integrating" independently derived lower level views. Some application areas may not even be planned for the next several years yet may be totally involved in specifying data requirements. The challenge is to include only the necessary level of detail in the Enterprise and Global level models.

5.2.2 State of the Art

The requirements phase of the system development process is composed of three activities - requirements collection, requirements specification and requirements analysis. These activities are conducted and accomplished by various individuals in the organization, including system users, analysts and designers. The characteristics of the individuals involved are a determining factor in the approach of the requirements phase and the techniques employed.

There are four major requirements approaches: top-down, bottom-up, backward-forward (output driven) and activity analysis (process driven).

A top-down approach views the organization as a whole and decomposes it by some predetermined criteria. This approach can be used to collect information and processing requirements either independently or simultaneously.

The bottom-up approach of gathering requirements is based on the assumption that modules or programs are the basic elements of any information processing system. The information processing system is assumed to grow in response to needs usually stated in terms of adding new processing requirements or programs. This bottom-up analysis is based solely on the definition of the existing or known processing requirements.

The backward-forward approach (output driven) is based on the input-process-output view of an information system. This approach is called backward to forward since it begins with the identification of outputs. For each process, the data utilized, both from inputs and internal data, are identified. This is usually accomplished in a top-down manner. First, the high-level output-process-input data stream is determined. Then the output is decomposed into its constituent data items and a backward flow is determined for these items. Subsequently, these data items are related to processes and both the processes and data are related to the process's source of this data requirement.

Activity analysis (process driven) is also based on the input-process-output model of an information system. The analysis begins with the identification of processes, both manual and automated. A process is synonymous with a system activity. For each process, the input(s) required to accomplish the process and the source of each input are determined. Additionally, the output(s) produced by the process and their destination are identified.

5.2.2.1 Requirements Collection

Before any analyst starts collecting requirements, a requirements collection plan for the system development effort is prepared. In this plan, the data to collect is determined, the potential sources for all data are identified, and a schedule for collection is prepared. The inputs to this activity are overall organization system plan(s), the underlying (requirements) models and organizational practices.

The requirements collection activity is allocated an amount of resources for its successful completion. Due to this resource restriction, all requirement data may not be collected nor all sources contacted. Trade-off must be made by the coordinator of the requirements collection activity. The effectiveness of this activity is dependent on the perspective of the requirements collectors (the analysts). The analysts must concentrate on determining what the ensuing system should do, not how this should be accomplished.

There are four major data collection techniques: (1) Reviewing documentation, (2) Observing the operating environment, (3) Administering questionnaires, and (4) Interviewing pertinent individuals. Table 5-1 summarizes the characteristics of these four techniques.

5.2.2.2 Requirements Specification and Analysis

Requirements specification is the activity of documenting the requirements in a precise and standard form. Requirements analysis is the activity of determining the completeness, consistency, correctness and validity of the requirements documents. The type(s) and method(s) of analysis is dependent on the method(s) of specification.

There are two types of techniques to accomplish these requirements activities: manual and computer-aided.

5.2.2.2.1 Manual Techniques

In manual methods, the analyst collects the data required, collates, continually organizes and analyses the data, and then produces reports [TRH]. The data is documented in a combination of prose, tables, diagrams, flowcharts and decision tables. The basic tools of these methods are often only pencil, paper and forms. Usually, the analyst is required to spend a large amount of time doing clerical tasks.

There are basically two categories of manual techniques. The first category is prose oriented utilizing both natural and formal language, and charts. Examples of this category are the AUXCO method [Aux], Accurately Defined System (ADS) promulgated by NCR, and Analysis Requirements determination, Design and development, Implementation and evaluation (ARDI) [Cou, HMT]. The second category is graphically oriented techniques. In these techniques, requirements are documented pictorially and prose is sparingly used to augment these diagrams. Examples of the graphical techniques are Hierarchical Input Process Output (HIPO) [Jo,Ktzmn]; Structured Analysis and Design Technique (SADT) [R,RB,RS]; and Data Flow Diagrams (DFD) [GS]. The characteristics of these methods are summarized in Table 5-2.

5.2.2.2.2 Computer-aided Techniques

There are two categories of computer-aided techniques. In the first type, the computer is used as a mechanism to facilitate the use of an existing manual technique. Here the computer is used in place of the human to accomplish many of the clerical tasks and some of the analysis.

In the second type, new techniques are designed to make optimal use of the computer's capabilities. A technique of this type consists of three basic components:

TABLE 5-1
DATA COLLECTION TECHNIQUES

Technique	Input	Comments
Review Existing Documentation	System documentation: Management overviews System output Functional Specification Organizational documentation: Organization Charts Job Descriptions Policies	Good as first step in collection. Can identify other appropriate techniques and their scope.
Observe Operating Environment	Activities of individuals related to system operation Document handling Informal communications	Observer may have biases or influence processes observed. Limited to a small number of activities. Observation skills are not easily learned.
Questionnaires	Written responses to a set of questions from a large number of persons	Questionnaire design can influence its validity and worth of information collected. Questionnaire responses must be brief, easily recorded, and unambiguous. A follow-up can increase returns.
Interviews	Oral responses to questions from a selected sample of persons	Sampling allows quicker and more economical collection of data. Interview must be carefully planned and skillfully executed. As a follow-up, interviewee should have opportunity to review and comment on written interview summary.

TABLE 5-2
MANUAL TECHNIQUES FOR SPECIFICATION AND ANALYSIS

Technique	Description	Tools/Techniques	Reference
ARDI	System development is described as four phases: Analysis Requirements determination Design and development Implementation and evaluation	PERT charts Documentation standards for each phase Project Management Techniques	Hartman et al
HIPO	Graphical design aid and documentation technique. Decomposed system is documented in terms of inputs, processes, and outputs.	Charts: Visual Table of Contents; Detailed HIPO diagrams Coding pads, templates, manuals	IBM
SADT	Techniques for top-down, structured approach to requirements documentation, project planning, managing and evolution	Graphic documentation technique: activity and data diagrams Definition of personnel roles	Softech, Inc.
SA	Top-down techniques, using directed graphs, in which processing and information requirements are integrated and collected simultaneously.	Data flow diagrams	Gane and Sarson

a.) A language for stating requirements which is appropriate for the user and the analyst, and, at the same time, sufficiently structured for computer processing and analysis.

b.) A software package which will store, analyze, retrieve and display the information recorded in the language.

c.) A database for storing the requirements in a form that facilitates analysis and the presentation of information required by the analyst.

The techniques differ with respect to the syntax and scope of the structured language and the capabilities and reports generated by the software.

There are a number of computer-aided requirements techniques; Computer-Aided Design of Information Systems (CADIS) developed at Royal Institute of Technology of Stockholm, Sweden[BK]; Computer-Aided Systems Construction and Documentation Environment (CASCADE) developed at the University of Trondheim, Norway[ASS]; Computer-Aided Design and Evaluation System (CADES) developed by International Computer Limited[ICL]; Problem Statement Language/ Problem Statement Analyzer (PSL/PSA) developed by the ISDOS Project at the University of Michigan[ISD1,ISD2]; and IDEF (ICAM Definition Method) developed for the U.S. Air Force by Softech, Inc. and Hughes Aircraft Company[BMRSP]. The characteristics of these methods are summarized in Table 5-3.

When any organization is considering the move from a manual to a computer-aided requirements technique, the costs and benefits of such a decision must be considered. The costs include technique acquisition, computer costs for installation and operation, personnel, personnel training plus others. The benefits include higher quality requirements documents, improved user involvement, improved coordination of the activities of the requirements collection and analysis process, better access to requirements, and improved system development with reduced time and costs.

5.2.3 Future Directions

Rapid progress in the requirements phase of system development will depend on the development of precise, generally accepted terminology and the achievement of general goals which are relevant throughout requirements analysis.

5.2.3.1 Standard Terminology

The development of standard terminology is important in the entire database field; it is especially so in requirements

TABLE 5-3
COMPUTER-AIDED TECHNIQUES FOR SPECIFICATION AND ANALYSIS

Technique	Description	Computerized Features	Reference
CADES	Uses structural modeling technique to generate an information-oriented system design. Generates implementation code and test data from design specification.	Design-information database Implementation code database Automatic generation of code and test data.	International Computers, Ltd. [ICL]
PSL/PSA	Provides user with PSL, a structural language, to document what system requirements are. PSA programs are used to create, analyze and generate report from the data base of PSL specifications.	(All in PSA) Verify syntax and semantics of PSL statements. Create, modify and access design data base. Produce a variety of standard reports utilizing four modes of presentation: lists and tables, matrices, pictures, and prose.	Teichroew et al. [TRH]
IDEF	Uses forms-driven graphical technique to specify entity-relationship-attribute (ERA) models. Analysis enhanced through defined roles of analysis team members and specific analysis procedures.	ERA dictionary database and reporting programs	SofTech, Inc. and Hughes Aircraft [Sof]

analysis. For example, the terms "requirements" and "specifications" are sometimes used synonymously, and are sometimes used to describe general and detailed documents, respectively. A "requirements specification" may mean generality or details or something in between. Clearly, such differences in terminology are apt to lead to confusion and errors, or to cause a diversion of effort from other matters.

5.2.3.2 General Goals

The following are general goals for the requirements phase:

- Development of a comprehensive methodology for the collection and analysis of requirements in the context of a methodology for the whole information system design process.

- Development of techniques for recognizing errors as soon as possible.

- Development of incentives for ensuring that the planning phase is adequately performed.

A comprehensive methodology must specify precisely what outputs are to be produced at each step from planning through maintenance. The methodology must also describe how the quality of the plans, requirements, designs, etc. is to be verified and maintained. In the requirements phase of the methodology this is particularly important because of the difficulties in communicating with users, the subjective nature of the communication, and very high cost of any errors, incurred during later phases. The methodology should be adaptable to a variety of environments: simple or complex corporate structures, simple or complex information systems, sophisticated or naive users, minor revisions or completely new systems, etc. The methodology could be hierarchically structured, with the degree of detail selected to conform to the environment.

A productive use of the methodology will also require that the various techniques and models be much easier to use than they are at present - the systems analyst will be considering a much broader part of the information system life cycle, and will be less able to master intricate technical details. In addition, as discussed later, there are already many problems in personnel training. The limitations and assumptions of techniques and models need to be more clearly documented, terminology clarified, etc. It is also desirable to develop very simple techniques and models for direct operation by the user. Systems that are menu-driven, question-and-answer, etc., might be easy to use and productive, and provide feedback more quickly and reliably than a systems analyst. Later paragraphs discuss some of the goals which should be achieved to provide the technical basis for such systems.

Errors tend to be most costly when they occur early and are recognized late. This is particularly important in planning and requirements because errors in these phases tend to have wide-ranging effects in later phases, and because errors are frequently unrecognized until the time when system integration, or testing, or even operation and maintenance occurs. The people who would recognize such errors - the users and managers - are simply not involved during design and implementation. For example, if the objects of interest are incorrectly identified during the development of the Enterprise Requirements model, then later assignment of data elements, construction of physical database design, etc., can do little to satisfy the real requirements. Current practice, unfortunately, is to devote a majority of the resources to such details as data element assignment, to the detriment of the more critical earlier activities.

Errors in planning tend to be extremely costly, since they affect many requirements, which in turn affect many designs, programs, etc. However, planning is rarely adequate, either because its importance is unrecognized, or because it is a high-visibility cost with no easily quantifiable benefit, or because the tools and methodology for adequate planning are unknown or nonexistent. The importance of an overall information system design methodology must again be stressed, because that methodology should provide the definition and justification for the products of the planning phase.

5.2.3.3 Goals in Requirements Collection

The following goals are particular to the collection of requirements:

- Development of methods for training personnel in requirements analysis.
- Development of techniques which are independent of the models used in later phases of database design.
- Development of tools and techniques for reducing naming problems.

Training is particularly difficult because personnel must have not only the technical skills to understand both computer technology and business practices, but also the proper perspective to avoid becoming enmeshed in unnecessary details, particularly when building the Enterprise Requirements and Global Information models. At present, training seems to be primarily confined to unstructured apprenticeship on the job, and is long and frequently fruitless. A means for assessing the potential ability of possible trainees would be very desirable - this might include psychological profiles, aptitude tests, and evaluation of previous experience. Aptitude and extended experience in programming, for example, may cause problems for a trainee who has to unlearn previous techniques and perspective. Underdeveloped interpersonal skills, particularly diplomacy in dealing with non-technical people, may also cause problems for former programmers. Personnel with extensive business experience, on the other hand, may not be receptive to requirements which do not conform in their views of how things should be done. In either case, a satisfactory methodology is needed to provide guidance, perspective, and a means for recognizing and correcting errors during the collection of requirements.

Current collection techniques seem to be quite dependent on models used in later phases of database design. This is understandable, because requirements collection often involves the review of tremendous volumes of data of little relevance - it is very tempting to look toward later phases and collect only the data needed for a particular model. This may cause future problems if the model is changed; requirements collection must begin again, with attendant increases in project cost and time, and deteriorated project status and morale. This again emphasizes the importance of a methodology, particularly one which supports a variety of models.

Naming problems may also cause costly iterations of requirements collection, particularly if the Enterprise Requirements and Global Information models have not been developed satisfactorily. Different terminology for the same thing, and the same terminology for different things, should be recognized and resolved as soon as possible, preferably while requirements are being collected. This is a very time consuming process, particularly at the level of individual data elements; databases with thousands of data elements are not uncommon. At this detailed level, the Global Information model provides a very useful way of categorizing data elements, so that only a small number must be compared with each other; clearly, errors in the Global Information model are very likely to lead to many more errors in the assignment of data elements.

5.2.3.4 Goals in Requirements Specification and Analysis

The following goals are particular to the specification and analysis of requirements:

- Development of "common sense" analyses.

- Development of techniques for comparing the results of different phases.

- Development of techniques for determining flexibility or "robustness".

- Development of a mechanism for easily making and propagating changes.

"Common sense" analyses are generally not particularly sophisticated, but they are extremely valuable. They may be used, for example, to suggest requirements which the user has failed to express because they are too obvious to be recognized consciously. These analyses are learned through experience and are generally not well documented, so they contribute substantially to the length, difficulty, and uncertainty of personnel training. Two types of common sense analyses are particularly important: heuristics or rules of thumb based on human factors, business principles, etc., and the deduction of consequences of requirements. The heuristics may suggest that a user is asking for too much data, or is imposing unnecesarily short response time, or is ignoring an important segment of the application. Such analyses are heuristics, rather than reliable procedures, because they involve matching a particular situation against an imprecisely defined ideal model. For example, we know that inventory control involves stock items, levels, prices, warehouses, purchase orders, etc., which are related in reasonably predictable ways.

The requirements analyst must be able to deduce reasonable consequences of requirements, in order to provide effective feedback to the user. For example, the restriction of each salesperson to a single warehouse can greatly simplify a database, at the expense of making it difficult to change to a future environment of multiple warehouses. Without guidance from the system analyst, the user would have no reliable way of judging the consequences of arbitrary, erroneous, or changeable requirements.

A closely related goal is the development of techniques for comparing the results of different phases. For example, the Enterprise Requirements Model could be compared with an application area requirements model to ensure that there was no conflict. This appears to require the development of rather complex mappings from one model to another. It is by no means obvious that there are non-trivial mappings which would apply to a large number of different organizations; it may be desirable to have organization-dependent variations of general mappings.

Another goal is that of developing techniques for determining flexibility or "robustness." The long range Business Plan, the Enterprise Requirements Model, and the Global Information Model all help to make requirements independent of one another, and hence more flexible. (Similarly, the logical database design provides independence for programs and the physical database.) It would be very desirable to have some objective measure of how much flexibility has been achieved, and what are the causes of a lack of flexibility. If this goal could be achieved, it would be possible to attain the general goal of recognizing and correcting errors at a very early stage, before they become the basis for a large amount of detailed work.

A final goal is the development of a mechanism for easily making and propagating changes. Making and propagating changes within a particular model requires the development of fairly complex integrity constraints on the model representation, but would otherwise be reasonably straightforward. Making and propagating changes between models is very difficult, since this capability requires the previously mentioned mappings between models, and probably a man-machine dialogue.

5.2.4 The Role of the Data Dictionary System in Requirements Analysis

Data dictionary systems can play a central role in the requirements phase. As a repository for the information collected during the first phase of requirements analysis, the DDS can relieve the clerical burden of the collection activity. In the specification phase, the DDS can serve both as a tool and as a control mechanism. Finally, the dictionary can provide certain basic types of analyses and also serve as a database to which more sophisticated analytical tools may be applied.

5.2.4.1 Collection

The collection task of the requirements phase, whether in support of enterprise modeling or application modeling, is a process in which the analyst works with end users to determine and document the data requirements for the enterprise and/or for the application at hand. Determination of requirements is basically a human activity in which the role of the DDS is one of a passive repository for the documentation of requirements. Since many individual users and user groups may be involved in this task, the amount of information gathered initially may be quite voluminous. Use of the DDS can reduce the burden of compiling and cross-referencing this information manually. Further, if proper dictionary entry types, e.g. data elements, reports, screens, etc., are available the use of the DDS can have a standardizing effect on the efforts of several analysts, or of one analyst surveying several users.

If the data dictionary system is used as a documentation tool for documenting the enterprise model and existing applications, analysts can use the dictionary database for reference and direction during the collection phase. Data sources can be identified for data elements or data classes described in the dictionary. Existing databases which contain required data or applications which affect required data can be located. This store of information can aid the analyst in selecting users to interview or query and in identifying systems or databases which may have interfaces with the new application. The enterprise model contains the data class ASSETS and defines the business functions and organization units interested in assets, the requirements for information on TRANSPORTATION EQUIPMENT can be compared to those of other previously defined assets for completeness and consistency.

The DDS characteristic most necessary to support an analyst during collection is ease of use, both for recording newly collected data and for accessing existing documentation. The analyst should be able to manipulate dictionary contents easily and should be able to evoke query responses and reports tailored to one's needs. This means that the inquiry/reporting capability of the DDS must be highly selective. For example, the analyst should be able to look at only the names and definitions of data elements related to ASSETS, not be forced to ferret this information out of larger, more general reports.

5.2.4.2 Specification

Specification involves the recording of data collected on user requirements and usually follows a particular methodology. The constructs and rules of the methodology are designed to force the analyst to specify completely and unambiguously the data and processes required. The interim results of specification are reviewed with the users and suitable modifications are made when necessary. Also, the analyst will subject the requirements specifications to one's own analyses (the analysis phase) and recreate the specifications as a result. When complete, the specifications should provide a full picture of user requirements and should contain enough detail for logical database design.

The major contribution that a data dictionary system can make to specification is to support the primitives, e.g. object types and relationships, of the methodology being employed by the analyst for specification. For example, if the analyst is using the E-R model [Ch] as a specification tool, the DDS should be capable of supporting objects such as entities, attributes, and relationships. In addition to enabling the analyst to record characteristics of interest about these primitives, the DDS should also be able to exercise control over the capture of this information. For example, if 'NAME' and 'KEY' are two important attributes of object 'ENTITY', then the DDS should reject any instance of an ENTITY in which these attributes are not specified.

Since the output of specification is to be reviewed with users, the dictionary system should support forms of output suitable for this task. A variety of output modes, e.g. graphics, prose, tables, etc., should be available for the analyst to select the form most appropriate to his/her users.

Additional features that are of interest during specification are controls. Basic controls on names of objects and relationships, can identify redundancies and inconsistencies. Controls, specific to the methodology itself, are also desirable.

For example, in Structured Analysis (SA) [Ga] each dataflow must have a source process and a destination process. The DDS should check and enforce this requirement.

5.2.4.3 Analysis

In requirements analysis the analyst critically reviews the specification of requirements in order to reduce redundancy and ensure consistency. In addition he/she may wish to examine the interfaces between existing requirements/systems and the newly specified data needs. Questions of semantics and completeness must be raised and resolved.

While certain types of analysis tasks can be carried out in a programmed fashion most of the analysis is a human mental activity. Thus, the data dictionary system can serve best as an aid to the analyst rather than as a substitute for the analyst. The aid required is primarily data access and reports. As a minimum the DDS must be capable of generating cross-reference reports on the objects defined in the dictionary. It must also be able to generate where-used reports for any specific object defined. Further, it must be able to generate traces of the effects of change. For example, suppose the element used to identify an EMPLOYEE is changed from EMPLOYEE NAME to EMPLOYEE NUMBER. What is the impact on other entities, reports, input screens, etc. of such a change? Finally, the analyst must be able to specify selection criteria governing reports requested from the dictionary system. It should be possible to qualify requests by object type (e.g., all entities), by values of attributes, (e.g., all 'weekly' reports) and by very specific object identifiers, (e.g., data item = 'EMPLOYEE NAME.')

One type of basic analysis that does lend itself to programmed application and thus becomes a candidate for inclusion as a DDS facility is completeness checking. All definitions can be checked to ensure that required attributes are indeed recorded and all references to other object definitions can be verified. Further, any standards of consistency rules positioned by the specification methodology can be checked and verified.

5.2.4.4 The Development of the Enterprise Model

The development of the Enterprise Model involving the same phases as application requirements analysis, places additional demands on data dictionary system support due to the nature of the information represented in the model and the way in which it is intended to be used. Again, the DDS must support a variety of constructs, e.g. business processes, organizational units, etc. Since, during the development of the Enterprise Model, definitions will change and evolve, the DDS must be able to record and accommodate such evolution. Relationships among organizational units (users), business activities, and the data classes and subclasses of interest to the enterprise must also be captured. Finally, the analyst must be able to generate reports regarding this information in user-friendly formats.

To support both enterprise and application modeling, the dictionary system must be able to record information on objects at several levels of abstraction. For example, one must be able to represent the data class EMPLOYEE, the subclass SECRETARY, and the record type EMPLOYEE RECORD. Further, the DDS must be able to document the differences between these objects at different levels as well as the mapping(s) necessary to go from one level to another. Finally, the ability to check for consistency between levels as well as on within levels would be most desirable. For example, if the above named EMPLOYEE RECORD is defined as a part of the Secretarial Skills Inventory Application, are the attributes represented therein those of EMPLOYEE or of the subclass SECRETARY?

5.2.4.5 Summary

The features required of a data dictionary system to support the activities of requirements analysis can be summarized under three categories: definition, access (reporting), and control (see Table 5-4). Unfortunately most data dictionary systems have been developed with support for DBMS data definitions as the primary objective. Currently none provide the full range of flexibility and analytic capability necessary for the requirements analysis task. Existing packages include, at best, some reporting features, limited support for non-standard object definitions, and some basic controls such as checking for duplicate object names.

Further research and development is required on more flexible and extensive definitional capabilities, on user-oriented modes of output (such as graphics), and on more extensive, perhaps user-specified, controls. These improvements would make the data dictionary system a valuable part of existing and future requirements analysis methodologies.

TABLE 5-4
DATA DICTIONARY SYSTEM CHARACTERISTICS AND CAPABILITIES NEEDED TO SUPPORT THE REQUIREMENTS PHASE

Characteristics/Capabilities	Relevant Task(s) in Requirements Phase			
	Collection	Specification	Analysis	Enterprise Modeling
Definition:				
Support for a variety of information and processing model(s) primitives.	X	X		
Support for several levels of abstraction	X			X
Tailoring to a specific system development methodology	X	X		
Access:				
Ease of use	X			X
Selectivity of output requests	X		X	
Cross-reference and where-used reports		X	X	
Variety of output modes		X		X
Control:				
Enforcement of naming standards	X		X	
Verification/enforcement of methodology-related rules	X		X	
Controlled evolution of definitions	X	X		X

5.3 INFORMATION MODELING

5.3.1 Introduction

The working group on information modeling within this panel addressed three major issues:

- * The Information Modeling Process. The group identified the basic processes involved in information modeling: user application modeling, view integration, and process specification. This led to an analysis of desirable features for an information model. Several information models may be considered as candidates for information modeling (e.g. [SS], [Ch], [SSW], [Co2], [NS], [SLo], [HM1], [EW], [F], [HWY], [HK], [Sh]. We have also included several references in Section 5-6 that give an exhaustive list of works related to these models.

- * The Database Workbench Concept. The concept of a database workbench was proposed as an environment to support the database design process. The data dictionary system component of the workbench is seen as a repository of metadata about the database being designed. The uses of metadata in the workbench are discussed, and several workbench tools are proposed.

- * Database Communication. A growing concern is the communication among heterogeneous databases. The organizational and technological factors influencing the distribution of the data resource are examined. Two approaches to database system communication are presented. The first is based on data restructuring and conversion, and the second proposes a federation of databases.

5.3.2 The Information Modeling Process

The information modeling process takes as input the enterprise requirements specification. The goal of information modeling is to obtain an integrated, formal, implementation-independent specification (the information schema) of application-specific enterprise information.

The steps of the database design process are summarized in Table 5-5. Steps 4 through 7 correspond to information modeling.

The specification is integrated in that it is the product of the view integration process wherein the requirements of functional organizational units are reconciled and integrated into the global information schema. The specification is formal in

TABLE 5-5. Action Table for Database Design

--SPECTRUM--

USER --------- DATABASE ADMINISTRATION --------- TECHNICAL SUPPORT

1: Identify user groups.

2: Collect requirements.

3: Structure requirements into enterprise model and application models.

4: Integrate application models into an information model.

5: Identify data ownership.

6: Correct models.

7: Make integrated database schema available for review.

8: Define database subschemas based on knowledge gained during integration.

9: Propose implementation alternatives (DBMS types or file-based alternatives).

10: Apply transactions to the subschemas and estimate relation and connection access frequencies.

11: Compute aggregate load applied to the integrated database schema.

12. Design promising implementations in detail.

13: Identify response time constraints

14: Apply critical functions to the schema for response time assessment.

15. Compute aggregate performance and cost factors.

16. Select a design that appears viable.

17. Compute performance where response time is critical.

18: Check if selection and response times found satisfactory.

19: --- n : Iterate !

USER --------- DATABASE ADMINISTRATION --------- TECHNICAL SUPPORT

--SPECTRUM--

that the structural and operational semantics of the information model are precise and well understood. The information model should represent enterprise concepts, structures, operations, and constraints. Lastly, the information model is implementation-independent in that mappings should be provided to map the information schema into a database schema supported by a data model as implemented on a particular database management system (e.g., the relational model [Co1,Co2] as implemented in System R [A], INGRES [SWKH], the CODASYL model as implemented in IDMS [Cu] or SEED [G2], or the hierarchical model as implemented on IMS [IBM] or System 2000 [MRI]).

5.3.2.1 Desirable Features of an Information Model

First and foremost, an information model should provide a collection of semantic constructs to aid the database administrator in modeling the database-specific portion of the enterprise model that results from the requirements analysis phase.

By semantic constructs we mean the structures, operations and constraints available to specify the semantics of the enterprise as it relates to the database schema that will support user applications. A minimal requirement is that the information model support the notions of aggregation and generalization [SS, MS]. Further, the model should have a sound mathematical foundation as well as a collection of "orthogonal" concepts, that is, the modeling constructs should be free of overlap so that a real-world concept will have a natural and unique representation in the model.

Most existing models support the notions of data type (domains), entity sets, associations, properties of both entities and associations, and subtyping (generalization). Some models include the notion of time with the event concept. Some may include time at the enterprise level where events and procedures [T] are specified. These events and procedures map to triggers and transactions at the information model level.

Since the role of the information model is to represent user applications and integrate them into a global conceptual schema, the model should provide view integration mechanisms. Moreover, to effectively map the information model to logical and physical database structures, the specification of processing requirements is essential. Both view integration and process specification will be discussed shortly.

Finally, the information model should be implementation independent in that its concepts address the modeling of the "real world" rather than being directed toward the logical or physical structures of a target database management system. Therefore, mappings to user views (external schemas), logical structures, and physical structures should be provided.

5.3.2.1.1 View Integration

Database design is an extremely complex task. Thus, it is essential that the problem be subdivided into manageable parts to reduce complexity and facilitate staged implementation. Usually, such a partitioning is performed by application areas (e.g., accounting, marketing, finance, etc.). This gives rise to the notion of "local views." The view integration process consists of combining local views into a consistent global view (conceptual schema). Combining these local views involves resolving name conflicts (synonyms and homonyms), removing redundant relationships, combining entities, and identifying and resolving insertion, deletion, and modification anomalies.

Data models based on mathematical functions are well suited for view integration. This is because all data are represented in their most elementary form and are not biased toward a particular design. Also, the view integration process is conceptually simple in that it involves the combining of subgraphs according to a set of formal rules and heuristics.

In the Conceptual Data Model (CDM) ([HWY], [YWH]), the notion of "compatible sets" and "underlying value sets" is useful for view integration. Two sets are said to be "compatible" if they have the same underlying value set. For example, the sets CLERKS and SECRETARIES are compatible because they have the same value set, EMPLOYEE-NO. The underlying value set may not be declared explicitly as a set in a schema, but an underlying value set name is defined for each defined set. The concept of compatible sets permits nodes (sets) in different views to be either merged or "connected" via identical value functions.

View integration must be considered an interactive process. Sometimes the designer must be prompted for additional information. For example, defining an identical value function between CLERKS and SECRETARIES only makes sense if some employees can serve both as clerk and secretary. Operators must be provided to allow the designer to merge nodes, remove redundant functions, and input additional assertions. Of course, the data dictionary system is required to store the information and keep track of all design decisions.

5.3.2.1.2. Process Requirement Specification Support

Processing information is needed in the design process to ensure correctness and efficiency of the design. Approaches that are based solely on semantic structure information are unable to estimate the processing requirements of a design. Conversely, however, designs based strictly on processing considerations tend to be flexible. Modeling of processes in the context of a well formed semantic model is useful for:

- Determining inconsistencies in the integrity assertions (i.e., update operations are found to violate the integrity rules stated in the information model).

- Determining relative access frequencies along functional paths. This information is useful to the physical design phase.

- Identifying deficiencies in the semantic model. Given a large number of entity types, there are a vast number of potential nonfunctional relationships (e.g., many to many, n-ary relationships) and subtypes (or generalizations) that could be defined. The exercise of modeling the key processes aids in identifying additional schema constructs that are needed, and existing relationships in the model that are nonessential.

- Providing formal specification for detailed application development. After the design has been approved, the process specification serves as a specification for application software developers.

For any process modeling language to be useful in database design, it is essential that it provide constructs that obviate irrelevant details (with respect to logical database design). Furthermore, varying degrees of procedurality should be allowed to accommodate different levels of modeling (e.g., high-level specification of processing intent, versus procedural navigation through the schema).

5.3.3. Tools for Database Design

Once an organization decides that data is a resource to be managed, and adopts database management system technology, it is faced with the problem of effectively managing the database system life cycle. The life cycle [TF] can be divided into two phases: 1) application development consisting of requirements analysis and specification, logical design and physical design, and 2) database operation consisting of implementation, operation, tuning, and adaptation.

In the above section we discussed models to deal with the specification of the information and processing relevant to an organization. A database workbench environment is needed to specify, design, develop, test, and tune database-intensive applications. The workbench should be viewed as a collection of independent yet communicating tools. Each tool should be designed for a specific part of the life cycle and should have the capability of communicating with other tools. This communication is important in supporting the mappings between the levels of the database design process depicted in Figure 5-1. The workbench would rely on a data dictionary system for inter-tool communication. Information common to all tools would be stored in the dictionary and could be used by the tools as well as by the database designers.

In the following sections we present a survey of existing tools, discuss the role of the data dictionary system in the workbench environment, and focus on the role of *metadata* - data about data - in the data dictionary system.

5.3.3.1. A Survey of Database Design Tools

In this section we consider some existing tools and some tools under development that have appeared in the literature. All of them address logical database design. Some also address physical database design.

A. *Katz and Wong*. In [KW, K] a method is presented for both logical and physical database design. Logical design uses a *design model* that is similar to the Entity/Relationship model [Ch]. The semantic objects of the design model are entity sets and their properties, associations, relationships, properties of relationships and value sets. These semantic constructs are formally mapped into a *design schema* that is a graph where each node represents an entity set or a value set, and each arc represents a function.

Integrity constraints are introduced by assigning properties to functions, e.g., total, partial, 1-to-1, and onto, as is done in the Functional Model. The design schema functions represent logical access paths, referred to as *access mappings*, that can be used to navigate among objects.

An *access schema* that represents the access paths to be supported by the storage structures of a DBMS is obtained from the design schema. The access schema serves as input to the physical design process. The approach is to obtain an implementation-oriented physical design that is *independent* of the target DBMS structures. A separate mapping provides the translation to DBMS-specific storage structures.

The physical design process uses the algebraic stucture associated with the access schema to produce an "implementation oriented" storage structure design. Let $f: A \rightarrow B$ denote a function in the access schema. The function may have one of four access properties: (1) *evaluated* - for each a in A, f(a) can be found without a complete scan of B, (2) *indexed* - for b in B, the inverse of b, $f^{-1}(b)$ can be found without a complete scan of A, (3) *clustered* - elements of $f^{-1}(b)$ are physically "close" to one another, and (4) *well-placed* - both a in A and f(a) in B are physically close so that the cost of accessing both is less than the cost of accessing them individually.

The method assumes that all functions in the access schema are at least evaluated. It takes advantage of the total ordering of the other properties to determine the "optimal" storage structure strategy. A crucial input to the algorithm is the access frequency of the access mappings. They are rank-ordered from highest to lowest. The approach is to label the arcs of the access schema with either "W", "C", or "I" (for well-placed, clustered, or indexed, respectively) such that the labeling is maximal, subject to the constraint that the labeling be conflict-free. Four labeling constraints lead to conflicts: cluster, placement, path, and implied. These conflicts are resolved by the replication of schema objects, and the degree of replication is controlled by the designer or database administrator. Katz [K] couches the labeling problem as an integer programming problem that is difficult to solve. A suboptimal solution is also provided. The mappings to relational and CODASYL physical schemas are discussed.

B. Gerritsen, Gambino, and Germano

The material summarized in this section is presented in [G1, GG, GGG]. Gerritsen [G1] describes an environment to aid in the design of CODASYL databases. It consists of several independent yet interrelated tools:

(1) DESIGNER takes as input a formal specification of information requirements expressed as queries in a hierarchical language and produces an integrated Data Structure diagram for the application-oriented views.

(2) DBD-DSS [GG] is a decision support system for physical database design. The diagrams of DESIGNER and access path frequencies are inputs to DBD-DSS. Other inputs to the model are access cost, storage cost, execution cost, and parameters related to the DBMS and the hardware configuration. DBD-DSS produces an initial feasible solution for the constrained optimization model that represents the design decisions and constraints. The designer can vary design parameters to improve the solution.

(3) Dynamic Restructuring is a tool that allows the restructuring of the database schema without bringing the database off-line. Restructuring is done by marking both programs and data with version or generation numbers.

Gerritsen suggests the integration of the tools so that output of DESIGNER becomes input to DBD-DSS, and that output of DBD-DSS becomes input to the File Definition Processor of the DBMS SEED [G2]. Also, the access path frequencies should be supplied automatically to DBD-DSS. Detailed access-frequency-calculation algorithms are discussed in [GGG]. Finally, the optimization model in DBD-DSS might be extended to include decisions related to indexing, data redundancy, set redundancy, area allocation, ordering of sets, and media assignment.

C. The Database Design System

The design system uses a Conceptual Data Model (CDM) [YWH] that is based on functions between entity sets, and a procedural language (TASL) to describe processing on CDM models. The system accepts a CDM specification consisting of several local views and their processing requirements, and integrates them to form one or more global models. These are then used to generate a schema compatible with the restrictions of a data model supported by a DBMS.

The system being implemented consists of four modules, the first two of which compile CDM and TASL, respectively, into internal formats for system manipulation. The next module *integrates* the local views into a single global view, the conceptual schema. It also acts as a dictionary for the conceptual schema. The fourth module generates a schema by combining nodes into records. These last two modules are interactive, and function as assistants to the designer.

The integration module contains several features to assist in the design process. Heuristics are used to limit the functions that the designer must examine to those apt to be redundant. It can retain multiple, related conceptual models. This allows the design to be "backed up" to some preceding point in the design and re-done. It also permits the evaluation of alternative conceptual schemas.

The major goal of the fourth module is to generate a schema with "good performance" in several stages through a high-level, multiple-stage evaluation process. For example, the first stage could determine the structure of the entry points, the second could combine nodes to form records, and a third could then deal with some form of record placement. The results of each stage could be evaluated by using (possibly) different cost functions. The structure of these stages and their cost functions is currently being researched.

D. DBDSGN for System R

The relational database management system System R has many ways of executing an SQL [Cham] statement. There may be many physical access paths to each relation (including scanning the entire relation, and auxiliary access paths, such as indexes), and the optimizer chooses the strategy for accessing relations which appears best. DBDSGN is a physical database design tool under development at IBM Research by Finkelstein, Schkolnick and Tiberio [ScT] which selects access structures that perform well for a given set of SQL statements. It combines optimizer-supplied evaluations to decide 1) how each relation should be ordered (or left unordered), and 2) which columns should be indexed. Its input include: the database schema (specified in the system catalogs), the queries and their frequencies, statistics about the database relations (which may come from the catalogs), storage space limit, and the number of solutions desired.

DBDSGN runs as an application program for System R. It extracts the cost for executing statements with each plausible access path directly from the System R optimizer. By using the optimizer's cost estimtes as a basis for a tool, two immediate advantages are obtained. First, the tool becomes independent of any optimizer improvements; an analytical model for the cost of performing a given statement based on the current knowledge of the strategy used by the optmizer would become invalid if the optimizer computations were altered. Second, we can guarantee that any proposed solution is one that the optimizer uses to its full advantage. In fact, the estimates that DBDSGN makes will correspond exactly to those made by System R when queries are compiled.

DBDSGN works as follows. It analyzes the basic structure of the input queries. From this analysis, it finds out which atomic costs it needs, corresponding to a small number of configurations of indexes. The cost of any other configuration can be derived from these atomic costs. A column elimination heuristic allows them to dismiss certain index candidates from further consideration. A controlled search on the space of realizable subsets of columns is performed, leading to the discovery of a set of good solutions.

E. An Integrated Database Design Aid

The goal of this project is to devise and implement a prototype Integrated Database Design Aid (IDDA) [M1], which allows non-database experts to directly develop, use, and maintain databases. The IDDA includes facilities for: (1) specifying and organizing databases and the operations that will be applied to them (logical design); (2) describing performance requirements and establishing the mapping (transformation) to a physical implemention; (3) the interconnection (federation) of databases. As such, the IDDA can serve as an environment for developing, using, and maintaining personal and office databases, as well as large-scale database systems.

One approach to obtaining an executable implementation for a database with a given conceptual schema would be to employ a transformational technique with a specific kind of transformation system, and a particular set of transformations for a given semantic database model. The transformation system would accept a semantic schema and associated performance estimates/requirements, and generate as output a physical implementation for the data structures and operations defined in the schema. A transformation system of this type is interactive, calling on a designer to assist in the stepwise process of refining the schema so that it eventually becomes executable.

F. The Database Designer's Workbench

It is a graphics-oriented decision support system for database design, providing designers with a convenient environment for creating database designs and experimenting with different design strategies [FT].

An integrated framework is postulated, permitting facile utilization of database design techniques and tools. Its philosophy is to employ several small design tools, rather than medium or large- sized tools. This lends flexibility to tool use and combination, and with each tool performing one well-defined subtask, overlapping of function is eliminated. In addition, database designers are able to design, experiment with, and use new tools of their own. The Database Designer's Workbench encourages this by relieving the designers of some of the details involved in developing user interfaces, input/output of design parameters, and storage specifications. The graphic interface allows the designers to see the various designs they are considering for their database throughout the entire design process. Since designs are easily modified graphically, designers are encouraged to produce and consider more candidate designs. Also, when the design group is progressing from one design phase to the next and is faced with a set of possible designs to choose from, the system's graphical nature can significantly aid in the heuristics of selection.

The Database Designer's Workbench is being developed by the Database Systems Research Group at the University of Michigan under contract from the United States Air Force. Presently, the user's manual and the design specifications are being completed. Future goals include having a minimal prototype running on MULTICS; a more extensive and powerful version will be implemented later. Approaches to logical design for the DBTG model have been developed and implemented previously in some Ph.D. dissertations at the University of Michigan [MI, IPT].

5.3.3.2 The Role of the Data Dictionary System in the Database Workbench

The group viewed the data dictionary system as playing an active and central role in the database workbench architecture.

Conceptually, it is the repository of the information needed by the workbench tools. Moreover it is the mechanism by which tools communicate with one another. Thus the DDS has as one of its components a meta-database about the database being designed.

We envision an expanded role for the data dictionary system, as compared with commercially available products. In addition to providing lexicons of data item definition, data typing, and synonyms, the dictionary system concept should be extended to provide the following services:

- Design Log. To effectively document the decisions made at the various design stages, the dictionary system must have a logging facility. This feature would record the various options considered and state the reasons for particular decisions.

  For example, user processing requirements on an entity set might dictate a subtype partitioning of that set. This fact should be recorded and be available for future reference should processing requirements change.

- Design Audit Trail. The complexity of database design leads to the decomposition of the design process into manageable phases, e.g., requirements analysis, enterprise modeling, information modeling and view integration, logical and physical structure design, and performance tuning. The phases are linked, however, by the mappings that transform objects, operations and constraints from one phase to the next.

  The dictionary system should provide an audit trail that records the mappings used together with relevant parameter values. This information could be used to characterize "workload" from level to level. Sensitivity analyses could be performed to determine the sensitivity of a design to the parameters available to the designer.

  The audit trail feature also allows high-level user views to be related to the application processes that manipulate database objects. Inverse mappings are also important, namely, given a database object, what processes act on it, from which user view(s), and with what frequency. Information of this type is crucial for view integration and for determining access path frequencies.

- Inter-tool Communication. Workbench tools must be able to communicate with one another. For example, the constraints specified in the information model should be made available to other tools such as a form system for consistency checking of input data, application development aids which ensure the semantic integrity of user controlled updates, query optimizers that use access path frequencies, etc.

* Metadata Management. The features mentioned above show that the prime function of our data dictionary system is the management of metadata. The importance of this function is addressed in the next section.

5.3.3.3 The Role of Metadata in the Data Dictionary System

One of the world's greatest database adminstrators recognized long ago the importance of metadata: "Knowledge is of two kinds. We know a subject ourselves or we know where we can find information upon it."[1]

5.3.3.3.1 The Uses of Metadata

The advantages of having metadata accessible to the user are almost self-evident. In the first place, in an environment containing several independent and accessible databases, a user may simply want to know in which database or file a given lexical descriptor is meaningful. Since the internal descriptors are often coded or highly abbreviated, and since the user may initially use a term different from that used in defining the database, some kind of lexical translation will be needed. Textual comments are also important.

Metadata should also describe structural relationships and should make them apparent to the user. A query issued in a high-level query language based on a misconception of structural relationships can often produce misleading results. Although attempts are often made, especially by "intelligent" query systems, to represent the database in a way that the user sees it, the extent to which a database schema may be transformed to suit the user's view is limited, simply because the underlying "world model" of the designer and implementor may differ. For example, a relationship may be one-many in the database while the user thinks that it is one-one; there may be no direct relationship between two entities when the user thinks there is one.

At a lower level, data formats, report formats and the characteristics of system interfaces, should also be considered as metadata and will be useful for those who wish to effect data transfer between various systems.

---

[1] Samuel Johnson (1775), quoted in Boswell's Johnson.

Finally, metadata may also be exploited by other programs. High-level query systems and natural language systems both exploit metadata for navigational purposes within a database. It is usually the case that this metadata must be manually constructed for these systems, simply because it is not held in a clean structural form either within the database or as an adjunct to it.

5.3.3.3.2 Querying Metadata

Many interactive query systems allow for some form of query of metadata as well as of data. For ease of use it is desirable that the query language for metadata resemble the regular query language. However, for this to be successful, some representation of the metadata must be found within the data model itself. It was felt that the existing commonly used data models (relational, network, etc.) are not rich enough to permit this and that a more sophisticated "semantic" model is required before metadata may be concisely embedded within the database itself.

The complexity of metadata depends on the purpose for which it is used. For help in formulating queries, a relatively simple model is usually sufficient that informs the user of the terminology of the database, some informal description of the contents, and some more formal description of the relationships within the database (functional, many-many, etc.). For the purposes of update and understanding integrity constraints something much more sophisticated (and, as yet, poorly understood) is required.

An analogy with programming environments is appropriate here. The support systems for conventional compiled programming languages seldom provide any higher level description of the program, data types, subprograms, etc., than the program itself. Occasionally program libraries are built up, and some kind of modularity is enforced. However, these programming aids are generally held off-line in the accompanying documentation. The better interactive systems provide some method by means of which the programmer may ask questions about his environment: what are the names of variables, what subroutines are available, and so on. Examples are to be found in the programming environments of all the more popular interactive languages. The same will surely prove true for databases: on-line access to metadata will be of most importance in an interactive environment.

5.3.3.3.3 Operations on Metadata

This is one of the most difficult problems associated with the use of metadata. Should it be possible to operate upon metadata in order to update it or to produce new metadata? Some useful techniques may be conveniently described as operations on metadata, e.g., the generation of a user view. Accompanying this mapping there must be an associated transformation of database access primitives. If user views are to be constructed by operating upon metadata, the primitive operators should be well-defined, and some kind of calculus or algebra should be available for metadata operations.

The operators needed to construct user views are those that "lose" information from the database. More complicated (and even less well-understood) are operators that add information. There are a number of techniques that would also benefit from being formally presented as operations on metadata. They include constructing "superviews" [BM]: a view that integrates data in several independent databases; the related problem of merging databases; and the problem of dynamically extending the schema of an existing database to include new data. These are all problems that will require extensive research in database semantics for their resolution.

5.3.3.4 Desirable Workbench Tools

The group discussed what tools the workbench should have and came up with the following list.

A. Information Models

Existing information models provide semantic constructs, operations, and intrinsic constraints required to model both user and global information schemas. Although none of these models is machine-implemented at this time, we expect to see several operational in the next few years. Needless to say, the information model provides metadata to the data dictonary system, and is therefore of utmost importance to the workbench concept. The model should have mechanisms to support both view integration and process specification.

B. Mapping Tools

Support tools are needed for the various mappings in the design process: enterprise model (the result of requirements analysis) to information model, information model to access path schema and information model to logical and physical schemas.

C. Constraint Subsystem

The constraints defined in the Integrated Conceptual Database model can be implemented in four ways:

a. Some constraints are supported by the database implementation structure (i.e., ownership constraints are enforced by implementation hierarchies).

b. The remaining constraints may be stored as assertions, for interpretation at query processing time [SK].

c. Assertions may be automatically bound into access procedures during compilation phases that bind external schemas to the internal representation. Some experiments on automatic constraint procedure generation have been made, but no consistent theory exists.

d. If no assertion subsystem is available then the constraints should be included in procedural interfaces that are implicitly or explicitly invoked by programs using the database.

D. Query Optimization (and Cooperation)

Knowledge stored in the conceptual model can be profitably used in the processing of queries. For example: The model should document all enforced 1:N connections between relations so that the best join method can be chosen. The model must also document alternate paths to the result, so that the best access paths among logically equivalent paths can be determined.

E. Application Program Development Aids

The knowledge embodied in the schema can, even without automation, reduce the effort substantially during the design and planning phase for data acquisition for application programs. Furthermore, if the constraints can be used to automatically generate modules which check range, maintain the properties of relationships, update derived data, etc., then coding effort is reduced. But more importantly the reliability of the database, and the relevance of its contents will be greatly enhanced.

5.3.4 Database System Communication

In this section we deal with the problems associated with heterogeneous database system communication. The need for such communication may arise for two reasons:

(1) user requirements for ownerships and control of information may dictate a "distribution" of the information model, a different "data model" for each application area, and possibly the use of different DBMS and hardware configurations, and

(2) the systems may have been developed independently, are operational and must be linked together because of the overall enterprise information needs.

5.3.4.1 Organizational Factors of Distributed Databases

When enterprises move into database-oriented processing, an apparent conflict arises. The centralized collection of information implied by the database technology is often seen as the property of the enterprise. This notion is in conflict with organizational principles of delegation of responsibility and authority. We associate ownership of information with power and the ability to act but excessive centralization limits initiative and may sometimes fail to realize the full potential of information or lead to its misuse.

Hence we expect that most enterprises will choose to decentralize the database logically. Often, but not necessarily, physical distribution will follow. Management of corporate finances provides a simile: a single corporate account or multiple divisional accounts may provide the storage of credit, but individual managers have their own budgets and plans for revenue and expense. These budgets and plans can be compared to the application schemas obtained from the individual users. If the enterprise management delegates operational authority and responsibility to one of its members, then that member should have the corresponding data authority and responsibility.

Data ownership is the term used to describe a user's relationship to the data for which responsibility has been assumed and authority has been granted. If, for example, a bank branch manager is responsible for the bank's customers in the local area, then maintenance of the customer lists, their addresses, and the linkage to various accounts is the responsibility of this manager. The integrated schema may impose some constraints, for instance a customer with a non-zero balance may not be deleted, but the schema must also support all actions that are authorized, such as the entry of a new customer. The integrated schema provides the crucial information needed to coordinate the activities of the local schemas. Data ownership is noted in the information model. All data described must either be owned or be generatable from other data through the use of some computational procedure. These procedures are attached to the information model.

At times the read frequency at other sites in a distributed system may be so high that performance demands that locally owned data be stored at the read site. The data may then either be kept uniquely remote, or be replicated. If the data is replicated one site remains the primary copy site and is identified as such. If updates have to occur at more than one site, then more complicated mechanisms are required. The notion of a true-copy token [M] which identifies the primary version may be used to control such databases. The true-copy token resides where the update privilege resides, and is moved as needed. Before it is moved inter-site consistency is established. The movement of the true-copy token may be on demand or on schedule. An example of a scheduled movement of a primary copy is seen in the branch bank, where local accounts are maintained in the daytime, but central postings from other banks take place at night.

5.3.4.2 Technological Factors Influencing Distributed Databases

In some organizations the motivation to adopt database technology was questionable when it was found that expansion of existing functions, or integration of distinct applications has cost greater than expected, often greater than the cost of the original applications. The use of database technology - looked upon as a remedy to data processing ills - does not guarantee the required growth flexibility [GAO]. Many database systems even have a hard time living up to their existing specifications, since they are typically based on software of the late sixties or early seventies.

Distribution may involve distinct computers, but includes also distinct databases on a single machine. The logical problems are the same, although lower communication bandwidth to remote machines can make poor situations truly intolerable.

In order to plan appropriately for a logical integration of diverse applications and databases a substantial modeling effort may be required. Models, using consistent and formalizable notions, have to be established and verified for all significant existing tasks, using actual documentation or programmer knowledge. A high-level sketch, with unverified assumptions about actual operating processes, can only be misleading. In addition, models may be constructed for new intended applications.

An important result of the integration process is the knowledge that is collected about the data resources of the enterprise. Even if a database is not implemented as a result of this effort, the model provides managers, system analysts, and programmers with information about existence ownership, scope, and constraints of the data. If a continuing maintenance effort is made, then the timeliness and quality of the existing data may also be documented.

Operations integration will be more difficult. Only a few systems today manage distributed data. While algorithms for management of replicated data are well understood [GM] they do not appear in generalized systems. In many cases explicit bridges may have to be built by programming staff to share data among distinct users. These bridges may be specific, often moving files created for that purpose between systems. As alternatives we see: 1) automatic query processing throughout the distributed database (most methods discussed assume a global schema), 2) access through distributed transactions, and 3) sharing of files by copying between nodes.

The design of distributed transactions is critical for good response and consistency. Known query types are decomposed into component transaction sections, each transaction section is placed with the database it needs, some sections invoke and correlate information from the other sections. Careful planning can minimize problems of interference [BSR]. If consistent models are available in stored form on the participating databases, then remote queries can be phrased in terms of these queries and in a truly distributed fashion.

The Validity of Distribution

Many existing non-shared data are unintentionally distributed. The fact that existing databases were distributed indicates in most cases that the communication bandwidth requirements were indeed limited. Incremental integration is hence quite feasible. For instance, much of the data needed for the operation of a warehouse is used only locally. Information about shipments does not need to flow much faster than the shipments do. A small percentage of queries result in out-of-stock conditions and should be rapidly distributed over adjoining nodes. Management information for periodic distribution is best summarized locally, using globally established procedures, and can be transmitted at low priority.

Increased automation of operations, such as seen in modern warehouses, requires powerful and reliable local processing capabilities. Local systems may have multiple workstations. The demands on availability, control, and bandwidth make local storage essential. Many of these distributed systems will be relatively small. This will reduce some of the complexity now needed in database management systems designed to deal with extremely large files. Even though distributed systems will provide less sharing of computing power and less aggregate hardware utilization, there is a positive compensating factor due to the smaller scale of operations. In numerical terms for operations on databases that need sorting in some form, the n log n performance bound is less for m distributed systems holding n entities each (m(n log n)) than for a central system of equal size ((mn) log (mn)).

When systems become small they may need to depend to a greater extent on remote facilities for maintenance, logging, and recovery. Issues of communication failure and processor failure need detailed analysis. If data is replicated on foreign nodes, then some backup can be provided from those nodes.

5.3.4.3 The Data Restructuring and Conversion Approach to Database Communication

Normally, a distributed database system is viewed as the distribution of databases and database management function across multiple, tightly coupled nodes in a communications network. All nodes understand a common schema and a homogeneous set of commands as well as their related protocols.

In the realistic future, however, distributed data processing will be required among loosely coupled nodes that will communicate asynchronously. Furthermore, different nodes may use different DBMSs. To operate effectively in such an environment, three important facilities are required: a) a data dictionary system, b) a general data translation facility, and c) a store and forward communications service.

IBM's data extraction, processing, and restructuring system (XPRS) [IBM1] is one example of a general data translation facility. XPRS implements the DEFINE data definition language, and the CONVERT data restructuring language. Using data translation techniques, users can extract data from a source database and map it to a target database. The store and forward facility transports not only the data, but also the data translation program (e.g., DEFINE and CONVERT, respectively) that together provide a "data translation schema" of the data. The data dictionary plays the important role of relating the data translation schema to the database schema for the given source and target database management systems. In effect, the data translation and data dictionary facilities can serve as a "bridge" between heterogeneous DBMS's.

There are many scenarios for distributing the data mapping process. For example, files could be extracted from the source database, restructured in the source system, and then sent to the target system. Alternatively, if the data translation system resided in the target system, the unstructured source data could be sent (along with their definitions) to the target system for restructuring and insertion into the target database.

5.3.4.4 The Federated Approach to Database Communication

One of the most significant trends and important challenges of the next decade in the area of computerized database systems is an effective support of decentralized collections of data. Recent work on distributed databases has focused on techniques to store and access data at various nodes in a computer network, as if that data were part of a single logical database; these efforts therefore concern the physical distribution of data. Significantly, logical decentralization is an equally important concern, and the current approaches to logical database decentralization are inadequate. In response to the needs described above, a new database system (software) architecture has been developed, termed federated database systems. The goal of the federated database concept [MH, HM2] is the logical decentralization and partial integration of databases.

A federated database consists of a number of logical components, each having its own user-level structural specification (component schema). The components of such a federation are related but independent, and they may or may not be disjoint. Typically, a component corresponds to a collection of data needed by a particular user or application, or a collection of closely related applications. The components in a federation are tied together by one or more federal schemas that describe the data that is to be shared by the various federation components. (The federation notion can be applied recursively.) The federal controller is a database system functional module that supports communication and translation of data among the components of a federation, based on the federal schema(s).

At present, a database federation tool is being designed [MH], and a prototype will be implemented. This prototype tool will provide facilities to logically interconnect database schemas (component schemas); the tool will interact with a designer and accept a set of schemas that are to be merged into a federation.

An important use of the prototype federated database system development aid described above is in establishing controlled sharing among existing databases. To accomplish this, a semantic schema is defined for each component database, and the tool is then used to define a federal schema based on these component schemas. A prototype federal controller, an integral part of the development aid, will automatically support data communication and translation among the components, via the federal schema.

The last step in logically connecting the component databases is to develop a mapping from the actual database structure of each component to its corresponding semantic schema; this can either be done on a case-by-case basis, or mappings to generic classes of general-purpose database systems can be developed (e.g., to a relational database management system, to CODASYL-DBTG, etc.).

The federation concept has a much wider applicability than merely a method of supporting logical database decentralization: the technique of federating information is a form of abstraction that can be used for structuring both data and programs. When used as an abstraction technique, federation facilitates the construction of complex collections of data from simple collections of data, and the construction of complex program modules from simple modules. That is, data and program components can be combined into a federation. Moreover, the technique of federation abstraction can be applied recursively to form hierarchies and other complexes of data and programs.

5.4 INTERFACE OF LOGICAL AND PHYSICAL DATABASE DESIGN

In the following sections, relationships between logical and physical database design are identified. Some results of logical design, which influence physical design decisions, are reviewed and the impact of physical design decisions on logical database design is examined.

5.4.1. Results of Logical Database Design

Physical database design is based on some of the results of logical database design. Such results provide information about entities and their attributes, relationships between entities, and descriptions of operations that are to be performed.

For example, information about entities may include:

* occurrence volume, volume growth rate
* keys (identifiers)
* privacy and security constraints

For attributes of entities:

* storage requirements
* integrity, privacy, and security constraints
* number of distinct values, volume growth rate

For relationships between entities:

* owner, member entities
* number of member records per owner record, volume growth rate

For operations on entities and relationships:

* type of operation (retrieval, insertion, deletion, etc.)
* frequency of operation, frequency growth rate
* operation selection criteria, output attributes (if applicable)
* volume of records affected by operation, volume growth rate

These lists are further expanded and are discussed in more detail in the sections on physical databases.

In principle, logical design is accomplished without regard to performance, while physical design alternatives are transparent to users. In practice, however, transparency of physical design decisions is not always possible as the following sections show.

5.4.2 Overlap of Logical and Physical Design

Logical and physical database design for commercial DBMSs are not independent processes. To indicate their overlap, consider the simplified database design process of Figure 5-1 which consists of four phases. Logical design is identified with the first three phases; physical design is identified with the last two. The overlap occurs during the phase of Schema Design (SD, see Section 5.1).

The database schema in a specific data model identifies components of a database using the available schema constructs in that model. Typically these constructs are record types and relationships among record types. The next phase of realization of a schema supplies implementation details to the schema so outlined. Such details include specifying entry points or "CALC" records, file structures to be used, and whether or not parent pointers should be used in relationship implementations. The selection of specific methods of implementation is based on an analysis of logical access paths. During the final phase of database design the schema specification is used together with the knowledge of the transactions against the schema to perform a schema evaluation. An assessment of database performance is done to arrive at a proper assignment of physical alternatives.

Database design is an iterative process. Refinements to a schema are made, for example, by altering implementation details and reevaluating the schema. Those schemas whose performance is judged acceptable are candidates for actual implementation. This refining process is indicated by the arc at the bottom in Figure 5-1. The loop formed is often identified with physical database optimization.

Because commercial DBMS schemas are so closely related to physical database implementation, changing certain record type or relationship implementations causes schema designs to be altered. In Figure 5-1 the dotted arc from Schema Evaluation to DBMS Schema and to Global Information Model indicates restructuring which may sometimes be attributed to the overlap of logical and physical design. Examples of this overlap are presented in the following section.

5.4.3. Alterations of Logical Schemas for Physical Reasons

Although commercial DBMSs support a variety of methods for implementing files and linkages between files, not all fundamental methods are supported. By altering schema designs, such omissions are circumvented.

Consider the situation where record partitioning is not supported. Record partitioning, also known as segmentation, involves the partitioning of a record type by attributes to form two or more subrecord types. Record partitioning may be used for performance reasons (e.g., separating highly referenced attributes from those that are rarely referenced) or for security reasons (e.g., separating confidential information from public information). Record partitioning is achieved in commercial DBMSs by defining each subrecord type and interconnecting them by 1:1 relationships in a schema. Figure 5-3 shows the partitioning of an employee record type into primary and secondary subrecord types. Note that in addition to the alterations of a schema, application programmers must supply routines that are necessary to support operations on partitioned records.

Another example concerns file partitioning (i.e., the partitioning of a file into two or more subfiles). Like record partitioning, file partitioning is used for performance reasons (e.g., separating highly referenced records from those that are infrequently referenced) or for security reasons (e.g., separating confidential records from public records). File partitioning is achieved by defining a record type for each subfile in a schema (Fig. 5-4).

Some DBMSs do not support secondary indices. In such cases, an index for an attribute can be implemented by a record type, where each record of the type contains a distinct value that is assumed by that attribute. Each record is then linked to those data records that possess that attribute value. Figure 5-5 shows an inversion of the Department attribute of an Employee record type which is based on this construction. In a similar manner additional record types, which serve as indices to other "index" record types, can be used to construct multilevel indices. Thus, schema outlines are altered each time an index is created or removed.

Occasionally, DBMSs cannot handle large or variable length data records. To circumvent this problem, data records are expressed as a sequence of fixed length records. The first record of a sequence is the main record and the remaining are trailer records. The main records define a main record type and the trailer records define a trailer record type. The records of a sequence are associated by having the main record be the owner record and the trailer records as member records of a relationship occurrence (Fig. 5-6).

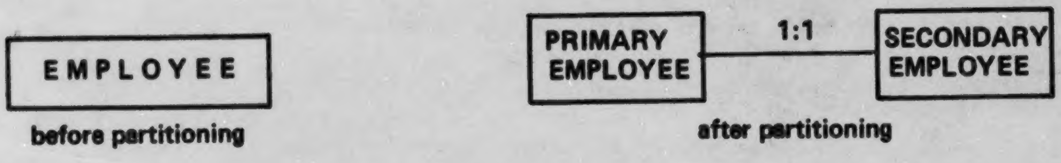
EMPLOYEE
before partitioning
PRIMARY EMPLOYEE
1:1
SECONDARY EMPLOYEE
after partitioning

Figure 5-3. An Example of Record Partitioning

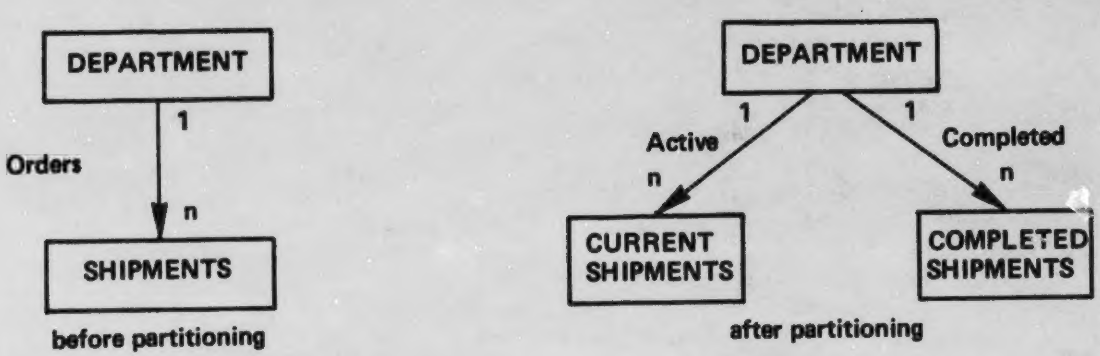
DEPARTMENT
1
Orders
n
SHIPMENTS
before partitioning
DEPARTMENT
1
Active
n
1
Completed
n
CURRENT SHIPMENTS
COMPLETED SHIPMENTS
after partitioning

Figure 5-4. An Example of File Partitioning

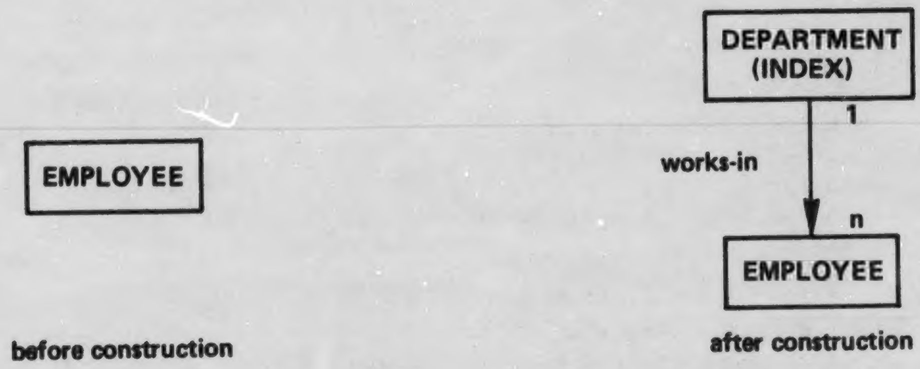
EMPLOYEE
before construction
DEPARTMENT (INDEX)
1
works-in
n
EMPLOYEE
after construction

Figure 5-5. An Example of Secondary Index Construction

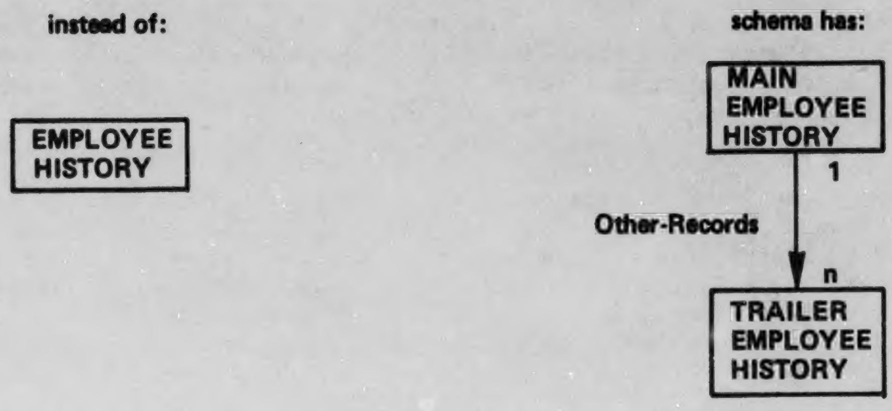
instead of:
EMPLOYEE HISTORY
schema has:
MAIN EMPLOYEE HISTORY
1
Other-Records
n
TRAILER EMPLOYEE HISTORY

Figure 5-6. An Example of Handling Large Records

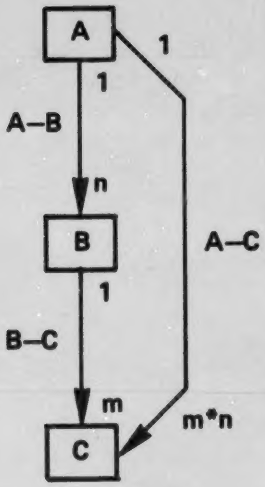
A
1
1
A–B
n
B
A–C
1
B–C
m
m*n
C

Figure 5-7. An Example of Redundant Relationships

A final example concerns the implementation of relationships. Relationships may be implemented by structures, such as pointer arrays and ring lists, or by procedures, such as joint operations. Using both methods allows almost all useful relationships to be defined in a schema outline. Only those relationships that are of primary importance to database processing need be implemented by structures; those that are not currently important could be implemented by procedures. Unfortunately, most DBMSs support only structural implementations, and consequently, alterations to schemas may occur.

Consider the case of Figure 5-7 where the relationship between record types A and C is redundant. That is, A-C can be deduced from the A-B and B-C relationships. For performance reasons, A-C might be implemented if database processing involves records of type A and C, but not B. In commercial systems, if A-C is implemented, then A-C is present in the schema. If it is not implemented, then it is absent from the schema.

The overlap of logical and physical database design is due primarily to the absence in commercial DBMSs of some fundamental methods of physical database implementation. If such fundamental alternatives (e.g. record or file partitioning) are available, the design process can be greatly alleviated. Another reason is that most DDLs used by the DBMSs to define schemas contain logical as well as physical constructs. Until DBMS DDLs are developed allowing a clear separation among the two types of design decisions, the logical/physical design overlap is likely to continue. Future DBMSs should strive to achieve the ideal of allowing the designer a clear choice among physical design alternatives which must be explicitly recorded and subject to modification for tuning purposes.

5.5 THE ROLE OF DATA DICTIONARY SYSTEMS IN MANAGEMENT OF DATA

5.5.1 Introduction

This section is a result of the discussions held by Group IV, which took a broader viewpoint with respect to the management of data and database design in a global organizational framework. They concentrated on the role data dictionary systems should play to facilitate data management.

Data is playing an increasingly significant role as our ability to technically exploit it is expanding through the use of computers and related technologies. This role will increase even more because of the growing dependence of organizations and managers on information systems for performing day-to-day operations. Today's managers are not only willing to use the computer, but expect to use it, and will be in a position to insist upon access to data to aid them in their jobs.

This pattern is likely to lead to the following:

1. Use of data by higher levels of management to aid in decision making will increase substantially.

2. End users will increasingly retrieve their data directly, obviating the need for programming by an MIS department.

A prudent person would expect this demand and prepare for it. Preparation requires increased emphasis on data with the management of data becoming a vital mission. In order to prepare for the anticipated rapid growth in DP, and for the increased use of data by end users, the time has arrived to address areas concerning overall management of data (in addition to the technical support of the DBMS itself).

In the past many organizations brought in a DBMS and then focused on the tools, standards, etc., needed to make the DBMS work. This frequently resulted in: 1) the data dictionary system being considered subordinate to the DBMS; 2) the organizations losing sight of the objective in bringing in the DBMS; and 3) the DBMS being used as an extension to file management systems' access methods.

What should have happened? The focus should have been (and still should be) on the result to be accomplished with a DBMS; that is, improved profits (or reduced expenses). "This may result from improved efficiencies or from better decisions through improved or more timely information."[Go] These, in turn, may be accomplished by improving data usability by managing data as a resource. Under this approach, emphasis is placed on the tools necessary to manage the data: the DBMS and the Data Dictionary System. The DDS is no longer subordinate to the DBMS. The DDS can and should be enhanced to provide for management of data.

First, what is meant by managing data as a resource? In general, it means increasing data usability through planning, organizing, directing, and controlling the data. Data management should make available a community of data to a community of users, across organizations, across functions, and across locations for people at all levels to aid them in performing their respective jobs.

Usable data is findable, understandable, available, organized, documented, maintainable, accurate, timely, private, and secure.

Data has the same characteristics of cost, value, and scarcity as the more familiar material, financial, and human resources [N]. Data is the raw material from which information is derived.

Similar to other resources, data must also be managed. As indicated by Bakke[Bak], activities necessary to manage resources in general (including data) are:

1. UNDERSTAND the resource - know thoroughly its nature and characteristics including its potential, limitations, composition, sources, and flow.
2. MAINTAIN and conserve the resource - acquire, maintain, and conserve quantity and quality of the resource needed by the organization.
3. EXPLOIT and employ the resource - develop and exploit the potential of the resource by its effective and efficient application to the activities of the organization.
4. INTEGRATE the resource - effectively integrate its use with the use of other resources to efficiently bring about a desired result.

5.5.2 Approach

The group approached the problem of determining the role of the data dictionary system by examining several specific areas of usage that may require DDS support. These areas included database design, distributed data, business systems planning, decision support systems, office automation, auditing, database administration, manual data handling, and the management of the metadata.

The following section contains a synthesis of the ideas brought up by examining the separate usage areas, but the report is not bounded by considering only traditional database applications. Rather, the role of the data dictionary system is approached by establishing what the DDS is and what it should be. The group's primary focus was on computerized data processing systems, so little was done to explicitly include non-automated data; however, the group attempted to avoid recommendations that explicitly, or by implication, excluded non-automated data.

5.5.3 Summary of the Recommendations

Though diverse applications were considered, it was decided that it is possible and desirable to have a single usage - independent data dictionary system to support the information management environment. The group views a DDS to be a central component of information management. With regard to the overall management of an organization's information, neither the DDS nor the DBMS should be considered subordinate. Rather, the DBMS and the DDS should be regarded as allied tools which are vital and virtually inseparable in accomplishing the common objective of managing the information resource. Furthermore, the key issue is not which one provides specific functions. Instead, the key issue is the functionality that needs to be provided collectively by the DBMS and the DDS. Such functionality is not provided by current systems, but is vital to manage data and make it really usable.

There are many types of data about databases that may reside either in the data dictionary or in a database controlled by the database manager. Regardless of its location, data about databases must be locatable through the DDS, and the dictionary system must associate the data about databases with other metadata in the environment.

A major group decision concerning the importance of data semantics led the group to include data semantics in the data dictionary system. Descriptions of the semantics of an organization's data are to exist in a form that is meaningful to a user and a form that is usable by the database manager as well.

The following outline summarizes the dictionary contents recommended by the group:

I. Administrative Data in the Dictionary

    A. Information Resource Management Data
    B. Descriptive Administration Data
    C. Processing Administration Data

II. Semantic Data in the Dictionary

III. Processing Data in the Dictionary

The advocated data dictionary system contains current DDS capabilities as a subset, but it is general enough to support future database systems where, for example, data semantics are incorporated into the database or the dictionary.

Defining The Data Dictionary System

Defining information management as a special instance of resource management, which requires understanding, maintenance, exploitation and integration of the resource, delimits the role of the dictionary system as being a primary and fundamental component of the overall data management function; and, does not constrain or categorize the data management activities supported by the DDS.

As a resource manager, a data dictionary system is

> an integrated repository that provides data necessary for managing data, where data management includes the planning,

control, direction, and organization of data. This metadata can be divided into three areas of functionality: administration, processing and semantics.

In essence, the DDS contains data that describes data. The data dictionary system is defined as "integrated" because it is a primary component of the information management system itself. This is in contrast to the more traditional view of the data dictionary system as a data management tool that is separate from (and subsidiary to) the database management system. Being an integral part of the database management environment implies that the DDS is the "bootstrap" database around which all other databases can be organized.

The remainder of this report concentrates on explaining the "areas of functionality" of the metadata. The functionality discussed below is not put forth as a complete definition, but is suggested as a good starting point for further investigation.

5.5.3.1. Administrative Data in the Dictionary

In the data dictionary, administrative data is data for the proper management of resources. The resources are either in the environment, or the resources are descriptions of administrative data or processes on administrative data which are the dictionary's own data. These three groups of data are outlined below:

A. Information Resource Management Data

1. Basic item description
2. Names, including synonyms, homonyms, and related names
3. Regulator(s), source(s), and sink(s) of the item
4. Business needs
5. User needs
6. Usage and frequency descriptions
7. Security constraints
8. Synchronization data
9. Integrity data
10. Recovery and resynchronization data
11. Audit trail

B. Descriptive Administration Data

1. Editing and Presentation formats
2. Encryption and report generation needs
3. Geographic location

C. Processing Administration Data

1. Security
2. Integrity

The following paragraphs explain the above components:

Information Resource Management data contains information necessary to manage not only the data but also the data management agent for the data and the user of the data. The data management agent may be an automated DBMS or an individual such as a clerk or secretary. In either case, the Information Resource Management data provides a global context within which the user or data management agent can understand the data item. The Descriptive Administration and Processing Administration data are the data dictionary's own description of what it contains: these two sections describe and abstract the description and processing data contained elsewhere in the dictionary (the description and processing facets of the dictionary system are described in the next two sections).

The following is a brief explanation of each piece of administrative data in the data dictionary.

A.1 The basic item description is a natural language explanation of what the item is, and its relevance to various applications, users and functions of the enterprise.

A.2 The names are those by which the item is known, as well as any name by which the item should not be known.

A.3 The regulators, sources, and sinks of the data item include the person, division, system, etc. that regulates the data item; the sources from which the data item is or can be received; and the sinks to which the data item is or can be directed.

A.4 The business needs document why the data was originally created and the purposes that it currently serves within the enterprise.

A.5 The user needs include user imposed constraints necessary for the data to be useful. Examples include the data item being absolutely current, or having been stable for at least a month, etc.

A.6 The usage and frequency descriptions provide information such as: "item must be processed daily", "use only after run xyz is complete", etc.

A.7 The security constraints provide a rationale for the security necessary for the item. An example would be explaining how federal law requires that this data item be accessible to a certain class of user, but not to another.

A.8 The synchronization data includes the current synchronization status of data as well as strategies for maintaining synchronization. An example would be a strategy of applying updates to an item that come from different sources in a particular order as well as indicators that the item is currently at a certain stage in the sequence defined by the strategy.

A.9 The integrity data includes rules and policies for maintaining data integrity.

A.10 The recovery and resynchronization data specifies how much and when to back up, contains procedures for activating back-up and subsequent resynchronization and details the granularity of recovery.

A.11 The audit trail contains procedures and time triggers to initiate the capture of auditing information, and implies the capability to trace any transaction from its source to output and vice versa.

B.1 The editing and presentation formats of the dictionary system's own data will vary by user, and may be significantly different from each other. In addition, active on-line control data must be properly formatted for the data manager, operating system, application program or the system being controlled.

B.2 The encryption and report generation needs are special processing requirements. E.g., the DDS data must be transported through the network in an encrypted format.

B.3 The geographic location of the DDS refers to the physical location of the metadata.

C.1 The security data profiles the permissible users and procedures for dictionary items.

C.2 The integrity data characterizes the procedures required to assure integrity for a dictionary item and to indicate how the procedures are invoked.

Items A.8 through A.11 are descriptions whose primary importance pertains to the run-time administration of the data item, whereas items A.1-A.7 are of primary importance in administering the use of data in database design. A.8-A.11 are considered to be "active" in the sense that they are on-line, used continually, and potentially subject to frequent update. Descriptions in B and C pertain to dictionary data itself, whereas descriptions in A pertain to data in the environment.

5.5.3.2 Semantic Data in the Dictionary

The data dictionary system should contain or point to the semantics of an enterprise's data, including the enterprise, organizational, and data structure models appropriate to supporting an enterprise's database applications. It should be the sole source of data definitions for all systems in the environment. In contrast to most of the administrative aspects of the DDS, the semantic data is intended to exist in a form that can be used directly by data processing applications and also be meaningful to the user. Therefore, the dictionary becomes the link to connect users and applications processes to information resources.

Data seen as characteristic of the semantic section of the data dictionary are outlined below:

A. Schema
B. Subschema(s)
C. Storage schema
D. DBA schema
E. Location in storage
F. Entity/attribute/relationship model (or record/item/set, etc.)

A. The schema is a description of the database as viewed by its users and as processed by a database manager (software).

B. The subschemas are compatible application views of a database.

C. The storage schema maps database data into a format to be stored on a memory device.

D. The DBA schema is the database manager's own view of the data under its stewardship.

E. The location in storage provides a description of the distribution of data items.

F. The entity/attribute/relationship model documents the enterprise view of data. The view may be stated in terms of entities, attributes and relations, records, items and sets, base relations or any other appropriate group of concepts.

Most of these dictionary items are traditionally viewed as a part of a database management system, but the group foresees a tighter interface between a Data Dictionary System and a DBMS. Whether a schema resides in the dictionary or in a database under the control of a DBMS is an implementation strategy that is immaterial to the task of the group. However, the DDS must "know" where the semantic information resides. In addition to making schemas available to users and DBMSs, the DDS must associate a schema with its users, applications and administrative procedure. So, regardless of implementation, the DDS and the database management system must be fully and functionally integrated.

5.5.3.3 Processing Data in the DDS

The Data Dictionary System's description of data should include the processes that can act on the data. The most significant departure of the group from the traditional view of a DDS is that the DDS contains processing information.

The processing information under the auspices of the dictionary system can range from the relatively mundane such as "use this hashing algorithm", to data semantics as represented by program segments (for example, "if a manager is requesting this data item, then summarize it and present the min, max, and average; but if a processing program is requesting this item, give it all the stored data").

Data characteristic of the processing aspects of the data dictionary system include:

A. Required inputs
B. Required outputs
C. Processing sequence
D. Alternative processes
E. Activation criteria for processing

A. The required inputs are the data items which are needed for a process to begin.

B. The required outputs are the data items which are produced as a result of a process.

C. The processing sequence lists the appropriate order in which to apply the processes.

D. The alternative processes provide a conditionally structured list of applicable processing to be used in case of exceptional events that may occur during the execution of a normal processing sequence.

E. The activation criteria are exceptional events which activate various processes: these are similar to the "on (condition)" constructs in languages like PL/I.

5.5.4 Dictionary System Implementation

The group did not directly address dictionary system implementation issues other than to state that functionality, not implementation, was important in considering the role of the DDS. For example, the DDS processing section could explicitly contain programs, or simply contain references to programs that are stored elsewhere: either way, the programs are a functional part of the DDS.

5.5.5 Conclusion

Group IV approached the definition of the role of a data dictionary system by first establishing what the dictionary system is. The previous sections outline this group's determination of the DDS as an integral and description-containing part of the database environment. As such the role of the DDS is inseparable from the role of the database manager. That role is best summarized as the planning, control, direction, and organization of the information resource.

5.6 REFERENCES

[A] Astrahan, M.M., et al., "System R: A Relational Approach to Database Management," TODS, 2:2, 97-137 (June 1976).

[ASS] Aadstadt, T.F., Schylstad, G., and Solvberg, "CASCADE - A Computer-Based Documentation System," Information System Analysis and Design, Bubenko, Lagefors, Solvberg (eds), Lund, Sweden, 1972.

[Aux] Auxtron Computer Enterprises, Inc. 2. Auxco Project Management System: User Reference Manual, 1972.

[Bac] Backus, J., "Can Programs Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs," CACM, 21:8, 613-641 (August 1978).

[Bak] Bakke, E.W., "The Human Resources Function", pp. 174-175, in E. Wight Bakke, Clark Kerr, and Charles W. Anrod, Unions, Management and the Public, second edition, (New York: Harcourt, Brace and Company, 1960).

[BCS] The British Computer Society, "Data Dictionary Systems Working Party Report," Data Base and SIGMOD Record, (1977).

[BF] Buneman, P., and Frankel, R.E., "FQL - A Functional Query Language," Proc. 1979 ACM-SIGMOD Conf., Boston, Mass., 52-57 (May 1979).

[BFM] Breutman, B., Falkenberg, E., and Mauer, R., "CSL: A Language for Defining Conceptual Schemas," *Data Base Architecture*, G. Bracchi and G.M. Nijssen, editors, North-Holland, Amsterdam, 237-256 (1979).

[BK] Bubenko, J. and Kollhammer, O., "CADIS - Computer-Aided Design of Information Systems, *Computer-Aided Information Systems Analysis and Design*, Bubenko, Lagefors, Solvberg (eds.), Lund, Sweden, 1975, pp. 119-140.

[BM] Buneman, P., and Motro, A., "Constructing Superviews," *Proc. 1981 ACM-SIGMOD* Conf., Ann Arbor, MI, 56-64, (May 1981).

[BMRSP] Brown, R., Morrison, R., Ramey, R., Sullivan, F., and Patrick, C., *Architects Manual, ICAM Definition Method*, (IDEF Version 1), Aug. 1979.

[Br] Brodie, M.L., *"Data Abstraction, Database and Conceptual Modeling: An Annotated Bibliography,"* NBS Special Publication 500-59, Washington, D.C. (May 1980).

[BSR] Bernstein, P.A., Shipman, D.W., and Rothnie, Jr., J.B., "Concurrency Control in a System for Distributed Databases," *TODS*, 5:1, 18-51 (March 1980).

[CB] "Proc. of the Conf. on Data Dictionary Systems," *Computer Bulletin*, 2:18 (December 1978).

[Ch] Chen, P.P.S., "The Entity-Relationship Model - Toward a Unified View of Data," *TODS*, 1:1, 9-36 (March 1976).

[Cham] Chamberlin, D.D., et. al., "SEQUEL 2: A Unified Approach to Data Definition, Manipulation, and Control," *IBM Journal of Res. and Dev.*, 20:2 (Nov. 1976).

[Co1] Codd, E.F., "A Relational Model of Data for Large Shared Data Banks," *CACM*, 13:6, 377-387 (June 1970).

[Co2] Codd, E.F., "Extending the Database Relational Model to Capture More Meaning," *TODS*, 4:4, 397-434 (December 1979).

[Cou] Cougar, Daniel, "Evolution of Business System Analysis Techniques," *Computing Surveys*, Vol. 5, No. 3, pp. 167-198 (September 1973).

[CP] Cardenas, A.F., and Pirahesh, M.H., "Data Base Communication in a Heterogeneous Data Base Management System Network," *Information Systems*, Vol. 5, 55-79 (1980).

[Cu] Cullinane Corporation, Integrated Database Management System (DMS) Brochure, 1975.

[EW] El-Masri, R., and Wiederhold, G., "Data Model Integration Using the Structural Model," *Proc. 1979 ACM-SIGMOD Conf.*, Boston, Mass., 192-202 (May 1979).

[F] Falkenberg, E., "Concepts for Modeling Information", in *Modeling in Database Management Systems*, North Holland, Amsterdam, 1976.

[FT] Fry, J.P., and Teorey T.J., "The Database Designer's Workbench," Users Manual, University of Michigan (1980).

[G1] Gerritsen, R., "Tools for the Automation of Database Design," *Proc. NYU Symposium on Database Design*, 91-97 (May 1978).

[G2] Gerritsen, R., "SEED Reference Manual," International Data Base Systems, Inc., Philadelphia (1978).

[Ga] Gane, C.T., "Data Design and Structured System Analysis," in *Tutorial and Software Design Techniques*, Third Edition, T. Freeman and A.I. Wasserman, (eds.) IEEE Computer Society, 1980.

[GACLP] Griffiths Selinger, P., Astrahan, M.M., Chamberlin, D.D., Lories, R.A., Price, T.G., "Access Path Selection in a Relational Database Management System," *Proc. 1979 ACM-SIGMOD Conf.*, Boston, Mass., 23-34 (May 1979).

[GAO] U.S. Gen. Acctng. Office, "Comptroller General of the U.S.: Data Base Management Systems--Without Careful Planning There can be Problems," Washington, D.C. FGMSD-79-35 (June 1979).

[GG] Gambino, T.J., and Gerritsen, R., "A Database Design Decision Support System," *3rd VLDB*, Tokyo, Japan (October 1977).

[GGG] Gerritsen, R., Gambino, T.J., and Germano, Jr., F., "Cost Effective Database Design: *An Integrated Model*," Working Paper 77-12-03, Dept. of Decision Sciences, Wharton, U. of Penn. (December 1977).

[GM] Garcia-Molina, H., "Distributed Database Coupling," *Proc. Third USA-JAPAN Computer Conf.*, San Francisco, 75-79 (September 1978).

[Go] Godlove, Richard H., "Impacting Management: Future Drections of DB and DP", *Infosystems*, June 1979, p. 68.

[GS] Gane, C.T. and Sarson, T., Structured Systems Analysis: Tools and Techniques, Prentice-Hall, 1979.

[He] Hecht, M.S., "Metadata Design for an Extended Relational Model of Data," unpublished manuscript, Bell Laboratories, Holmdel, NJ (1980).

[HK] Hecht, M.S., and Kerschberg, L., "Update Semantics for the Functional Data Model," Database Research Report #4, Bell Laboratories, Holmdel, NJ (1980).

[HM] Hammer, M., and McLeod, D., "The Semantic Data Model: A Modeling Mechanism for Data Base Applications," Proc. 1978 ACM-SIGMOD Conf., Austin, Texas, 26-36 (May 1978).

[HM1] Hammer, M. and McLeod, D., "Database Description with SDM: A Semantic Database Model," TODS, 6:3, 351-386 (September 1981).

[HM2] Hammer, M. and McLeod, D., "On the Architecture of Database Management Systems," Infotech State of the Art Report on Data Design (1980).

[HMT] Hartman, W., Matthes, H. and Troem A., Management Information Systems Handbook - ARDI, McGraw - Hill, 1968.

[HWY] Housel, B.C., Waddle, V., and Yao, S.B., "The Functional Dependency Model for Logical Database Design," 5th VLDB, Rio de Janeiro, Brazil, 194-208 (October 1979).

[IBM] IMS/VS publications, IBM, White Plains, NY (1978).

[IBM1] Data Extraction, Processing, and Restructuring System, IBM Manual SH20-2178 (1979).

[ICL] International Computers Ltd., "CADES - Computer-Aided Design and Evaluation System," General Information Manual (UK Software and Literature Distribution Center, ICL, ICL House, Putney, London SW15 ISW).

[ISD1] ISDOS Project, University of Michigan, Problem Statement Language, Language Reference Manual, PSA Version 4.2, May 1977.

[ISD2] ISDOS Project, University of Michigan, Problem Statement Analyzer, User Manual, Version 4.2, May 1977.

[IPT] Irani K.B., Purkayastha, S., and Teorey, T.J., "A Designer for DBMS-Processable Logical Database Structures," 5th VLDB, Rio de Janeiro, Brazil, 219-231 (October 1979).

[Jo] Jones, M.N., "HIPO for Developing Specifications," *Datamation*, March, 1976, p. 112-125.

[K] Katz, R.H., "Database Design and Translation for Multiple Data Models," Memorandum No. UCB/ERL M80/24, University of California at Berkeley (June 1980).

[KKT] Kerschberg, L., Klug, A., and Tsichritzis, D., "A Taxonomy of Data Models," in *Systems for Large Data Bases*, Lockemann, P.C. and Neuhold, E.J. (eds), North *Holland*, New York (1977).

[KOP] Kerschberg, L., Ozkarahan, E.A., and Pacheco, J.E.S., "A Synthetic English Query Language for a Relational Associative Processor," *Proc. 2nd Intl. Conf. on Software Engineering*, San Francisco, Calif., 505-519 (October 1976).

[KP] Kerschberg, L., and Pacheco, J.E.S., "A Functional Data Base Model," Technical Report, Pontificia Universidade Catolica do Rio de Janeiro, Rio de Janeiro, Brazil (February 1976).

[Ktzmn] Katzman, H. Jr., *System Design and Documentation: An Introduction to the HIPO Method*, Van Nostrand and Reinhold Pub. Co., 1976.

[KW] Katz, R.H., and Wong, E., "An Access Path Model for Physical Database Design," *Proc. 1980 ACM-SIGMOD Conf.*, Santa Monica, Calif., 22-29 (May 1980).

[Li] Lien, Y.E., "On the Semantics of the Entity-Relationship Model," *Proceedings of International Conference on the Entity-Relationship Approach to Systems Analysis and Design*, Los Angeles, CA (December 10-12, 1979).

[Lu] Lum, V.Y., et al., "1978 New Orleans Data Base Design Workshop Report," *5th VLDB*, Rio de Janeiro, Brazil, 328-339 (October 1979).

[M] Minoura, T., "Resilient Extended True-Copy Token Algorithm for Distributed Database Systems", Ph.D. thesis, Stanford University (1980).

[MBW] Mylopoulos, J., Bernstein, P.A., and Wong H.K.T., "A Language Facility for Designing Database-Intensive Applications," *TODS*, 5:2, 185-207 (June 1980).

[Mcl] McLeod, D., "A Database Transaction Specification Methodology for End-Users," Technical Report 79-6, Computer Science Department, University of Southern California, Los Angeles, CA (December 1979).

[Mc2] McLeod, D., "On Conceptual Database Modelling," to appear in *Proceedings of Workshop on Data Abstraction, Databases, and Conceptual Modelling*, Pingree Park, CO (June 23-26, 1980).

[MH] McLeod, D., and Heimbigner, D., "A Federated Architecture for Database Systems," *Proc. NCC*, AFIPS Press, Anaheim, California, 283-289 (1980).

[MI] Mitoma, M.F. and K.B. Irani, "Automatic Data Base Schema Design and Optimziation," *Proc. First VLDB Conf.*, Framingham, Mass. (September 1975).

[MK1] McLeod, D. and King, R., "Applying a Semantic Database Model," *Proceedings of International Conference on the Entity-Relationship Approach to Systems Analysis and Design*, Los Angeles, CA (December 10-12, 1979).

[MK2] McLeod, D. and King, R., "Semantic Database Models," to appear in *Principles of Database Design* (editor S.B. Yao), Prentice Hall (1981).

[MRI] MRI Systems Corp., *System 2000 Reference Manual*, Austin, TX (1978).

[MS] McLeod, D. and Smith, J.M., "Abstraction in Databases," *Proceedings of Workshop on Data Abstraction, Databases, and Conceptual Modelling*, Pingree Park, CO (June 23-26, 1980).

[N] Nolan, Richard L., *Managing the Data Resource Function*, West Publishing Co., 1974, pp. 3-4.

[Nav1] Navathe, S.B., "A Methodology for Generalized Database Restructuring", Ph.D. Dissertation, University of Michigan, 1976. (Available from University Microfilms, Ann Arbor, Mich., Order No. TSZ 7627, 577).

[Nav2] Navathe, S.B., "Schema Analysis for Database Restructuring", *TODS*, 5:2, pp. 157-184 (June 1980).

[NG] Navathe, S.B., and Gadgil, S., "A Methodology for View Integration in Logical Database Design," *Proc. 8th VLDB*, Mexico City (September 1982), to appear.

[NL] Navathe, S.B., and Lemke, J., "On the Implementation of a Conceptual Schema Model within a Three Level DBMS Architecture," *Proc. NCC*, AFIPS Press, New York, NY, 697-708 (1979).

[SLo] Su, S.Y.W., and Lo H.D., "A Semantic Association Model for Conceptual Database Design," *Proceedings of International Conference on the Entity-Relationship Approach to Systems Analysis and Design*, Los Angeles, CA, 147-171 (December 10-12, 1979).

[Sof] SofTech, Inc., *An Introduction to SADT - Structured Analysis and Design Techniques*, Waltham, Mass., Feb. 1976.

[SS] Smith, J.M., and Smith, D.C.P., "Database Abstractions: Aggregation and Generalization," *TODS*, 2:2, 105-133 (June 1977).

[SSW] Scheurermann, P., Schiffner, G., and Weber, H., "Abstraction Capabilities and Invariant Properties Modelling within the Entity-Relationship Approach," *Proceedings of International Conference on the Entity-Relationship Approach to Systems Analysis and Design*, Los Angeles, CA (December 10-12, 1979).

[StK] Stonebraker, M., and Keller, K., "Embedding Knowledge and Hypothetical Data Bases into a Data Base System," *Proc. 1980 ACM-SIGMOD Conf.*, Santa Monica, Calif., 58-66 (May 1980).

[SWKH] Stonebraker, M., Wong, E., Kreps, P., and Held, G., "The Design and Implementation of INGRES," *TODS*, 1:3, 189-222 (1976).

[T] Tsichritzis, D.C., "OFS: An Integrated Form Management System," *6th VLDB*, Montreal, Canada, 161-166 (October 1980).

[TF] Teorey, T.J., and Fry, J.P., "The Logical Record Access Approach to Database Design," *ACM Computing Surveys*, 12:2, 179-211 (June 1980).

[TRH] Teichroew, D., Rataj, E.J., and Hershey, E.A. III, "An Introduction to Computer-Aided Documentation of User Requirements for Computer-Based Information Processing Systems," ISDOS Working Paper No. 72, March 1973.

[W1] Wiederhold, G., *Database Design*, McGraw-Hill, New York (May 1977).

[W2] Wiederhold, G., "Management of Semantic Information for Databases," *Proc. Third USA-Japan Computer Conf.*, San Francisco, 192-197 (September 1978).

[YNW] Yao, S.B., Navathe, S.B., and Weldon, J.L., "An Integrated Approach to Database Design," in Database Design Techniques: Requirements and Logical Structures, in Lecture Notes in Computer Science, #132, (Yao, Navathe, Weldon, Kunii, eds.), Springer-Verlag, New York, 1982.

[YWH] Yao, S.B., Waddle, V., and Housel, B.C., "Database Design Using a Canonical Data Model and Process Specification Language," Purdue University Computer Science Report DB-80-10 (1980).

6. PHYSICAL DATABASE DESIGN

John K. Lyon

CHAIRMAN

Biographical Sketch

John K. Lyon is a consultant in General Electric's Corporate Information Systems Planning Operation in Bridgeport, CT. He is responsible for improving effective exploitation of database technology within operating components of GE, information systems business planning, and assisting components with specific information systems problems. Before rejoining GE in May, 1980, Mr. Lyon was Vice President of the data processing subsidiary of Colonial Penn Group in Philadelphia for four years, with responsibility for systems design and implementation, business planning, and effective utilization of technology. Prior to this, he had a fourteen year association with GE and Honeywell in defense, corporate research and development, computer marketing support, application and development, advanced database technology, and development of large business systems. Mr. Lyon is a graduate of the Pennsylvania State University and Union College, and is author of the Wiley publications "Introduction to Database Design" (1971) and "The Database Administrator" (1976).

PARTICIPANTS

Donald Batory
Thomas Baumgartner
Joseph Brownsmith
James Fry
Margaret A. Herrick
John K. Lyon
John Midgley
Joan Monia
Thomas E. Murray
Gary Sockut
Ruth Thorburn

6.1 INTRODUCTION

Database design is accomplished in two distinct activities: logical design and physical design. In simplistic terms, logical design is accomplished without regard to performance, while physical design alternatives are transparent to a user. Pragmatically, such alternatives are rarely transparent since they are generally reflected in the schema and impact the choice of DML (at least as seen by the programmer). The considerations of a DDS are independent of the data model used as the DBMS basis.

The Physical Design Working Committee addressed physical issues only in the context of the Data Dictionary System (DDS). Currently available (or projected) DDS's are incapable of systematically organizing data for all of the uses described below. This section does not indicate all of the uses of a DDS (e.g., DDL source generation, query language interpretation, etc.).

To describe the uses of the DDS in physical design the committee defined six physical database life cycle phases:

- User Requirements - response time, record occurrences, frequency of use, transactions, growth, privacy requirements, availability, logical design, etc.
- Physical Design - distribution, mapping, design validation (e.g., simulation, analysis) prototyping, design limits, recovery/restart, encoding, encryption, archiving, etc.
- Implementation - design confirmation, stress testing, data validation, load, etc.
- Operations - performance monitoring, maintaining integrity, housekeeping, etc.
- Evolution - application enrichment, reevaluation, usage shifts, migration, restructuring, reorganizing, etc.
- End-game - termination, shift to a static entity.

These six phases are not mutually exclusive in time and may reoccur throughout the life of a database.

For purposes of reference, four central concepts are defined:

Physical Database Design

Physical database design is the process of specifying and maintaining stored data and "database procedures." The primary goal of this process is performance efficiency within given constraints. Inputs to this process are:

- Anticipated database usage
- Available implementation techniques
- Hardware and software characteristics
- Physical characteristics of the data
- Logical data structure.

Decisions in the design process include

- Storage mapping
- Physical record size
- Access path selection
- Clustering/partitioning of data
- Encoding/compression/encryption of data
- Concurrency
- Restart/recovery.

Data Directory System(DDS)

A DDS is the repository of data defining/describing the entities (and their attributes) which constitute the information environment of the enterprise. Examples of such entities are: databases, files, records, data elements, software, users, reporters, transactions, security profiles, locations, schema, subschema, etc. In addition, a DDS contains data relative to the form and placement of data. Some key aspects of a DDS are:

- Its contents include data about data (metadata).
- The metadata supports database and file design decisions.
- Metadata is used by people and by systems.

Restructuring is any change in the logical design of the database. Examples:

- A changing from a one-to-many to a many-to-many relationship.
- Adding or deleting a data item.
- Reordering data items within a record type.
- Changing access privileges.

Reorganization is any change in the physical organization of the stored data. This should be invisible to end-users or application programmers, except via performance. Examples:

- Changing from hashing to indexing.
- Eliminating overflows in an ISAM file.
- Moving to a new physical device.
- Changing between data compression and non-compression.

Note that a change due to logical design will usually result in physical changes, while physical changes may or may not result in logical changes.

6.2 REQUIREMENTS - METADATA REQUIRED IN DDS

The DDS for a new database is initially created during the requirements definition phase using the logical design components (Entities, Elements, and Relationships) as its basic input data. During the life cycle of the database, data about the logical elements and their physical counterparts will be systematically captured and organized within the DDS. During this initial phase, data is collected from external sources (i.e., not from database operations). Such sources include user design specifications, design assumptions, statistical inferences from existing systems/files, etc.

During this phase of design, the emphasis is on how data will be used, how much data exist (or will exist), performance requirements, and operational environment. It is important to note that the data about data which are to be collected are the same as those requested in the non-DDS environment. The distinction involves recognition of the need to systematically organize the data in both human and machine readable form.

During the generation of the logical design (an input to physical design), metadata is captured about the four basic database components: entities, elements, relationships and subschemas (user perceived virtual records). The DDS must be capable of representing data about these components in human and machine useable forms. The metadata needed are:

Logical Data Entity

Occurrence volume
Growth rate addition: Frequency/Cycle
Growth rate deletion: Frequency/Cycle
Size (total of element sizes - can be generated)
Access privileges (e.g., read, add, delete)
Prime key(s)
Privacy (related to users)
Availability requirements (e.g., within two hours)
Constraining requirements (e.g., fastest response required)
Constraints (e.g., how long to keep) business and external

Logical Data Element (Attribute)

Size
Format(s) by frequency (may relate to operation)
Entity(s) described by element
Integrity rules (e.g., values 500-5000)
Privacy requirements (related to users)
Number of distinct values (size of domain) and growth rate (e.g., 1500)
Modification and selection operation(s)
Access privileges (e.g., read; modify)
Description of identifying characteristic of element (e.g., identifier/non-identifier)
Availability requirements (constraining requirement)
Constraint (business and external)

Logical Data Relationship

Owner
Member(s)
Average size of set/member type
Maximum size of set/member type
Minimum size of set/member type
Median size of set/member type
Frequency distribution type/count
Growth rate/member type
Addition of new entities
Deletion of entities
Modification of set occurrence membership

Logical Operation Against Virtual Record

Type (e.g., access, addition, deletion, modification):
  1) non-key, 2) key, 3) relationship key
Frequency per cycle
Processing mode (e.g., batch or interactive)
Response constraint (e.g., time) with priority—may vary with user
Sub-schema: entity and elements (attributes)
Access type (e.g., sequential; serial; random; relationship(s) — with sequence — with elements for sequence, no prime-key elements)
Purpose (e.g., to produce standard cost; to generate bank balance)
Number of virtual records expected in response

Notes:
1) All terms are used generically
2) Only factors which may be needed for physical database design included
3) Some factors will not affect physical database design with all DBMS's
4) Not all factors are developed to the same level of detail (e.g., privacy is not detailed, logical operation needs more development and checking of ideas)

6.3 PHYSICAL DATABASE DESIGN PHASE

There are numerous decisions and actions required to arrive at a physical database design which will efficiently support a logical database design. The basic objective of physical database design is physical grouping and placement of data which efficiently supports specified and projected data storage and retrieval while observing established requirements and constraints. It should be noted that there are usually several possible physical database designs which can satisfy the same logical database design.

Three major subphases of the physical database design process are identified below. The development subphase deals with the decisions and actions leading to a specific design. The allocation subphase addresses the distribution and placement of records within the available storage space. The validation subphase is concerned with specifying or accomplishing procedures leading toward validation of the physical database; i.e., does the design satisfy the requirements and observe the constraints.

6.3.1 Design Development

6.3.1.1 Distributed vs. Non-distributed. Affects design at highest level since, in a distributed system, all the physical aspects of a database are represented at each node of the network. Data needed for distributed design include the same data as needed for non-distributed database design and, in addition, include external requirements for data distribution:

- number of nodes in the network
- data description at each node
- size of database (record occurrences) at each node
- frequency data regarding the source and object of processing

6.3.1.2 Concurrency. This addresses the issues of two or more users accessing data simultaneously. These issues arise due to possible conflicts between users who are attempting to update or modify the same data. At the physical database design level, the concurrency issue may lead to data partitioning and data lock-out to avoid contextual conflict. Metadata needed to support analysis and decision making here are:

- user versus data usage
- mode of use (update vs. retrieval only)
- frequency of use

6.3.1.3 Storage/Access Methods. This area deals with the selection of basic methods for organizing, storing and retrieving of data. Data organizations include sequential, direct, indexed and linked. Many variations of these techniques exist and are in common usage. The type of data needed to support decisions in selecting storage/access methods include:

6.3.1.3.1 Storage frequency and type.

- insert/update
- sequencing requirements (e.g., by key value, within set)

6.3.1.3.2 Retrieval frequency and type.

- retrieval of records in a specified sequence
- retrieval of all or part of a set
- boolean operations

6.3.1.3.3 Type and frequency of anticipated changes to the logical data structure.

6.3.1.4 Selected Design Limits. This area deals with the establishment of "self-imposed" design limits, such as maximum numbers of record instances, a stipulated maximum degree of structured complexity, pointer sizes, maximum field sizes, key sizes, etc. The type of meta-data needed for this area would include:

- target database entity occurrences
- frequency and numbers of transactions or accesses by records or data classes
- estimated number of unique key values
- the ratio of database entities to defined relationships
- estimated storage requirements by database record type or data class

6.3.1.5 Hardware/Software Constraints. There may be numerous constraints imposed to reform the design of a physical database by the computer hardware and supporting system level software. The most obvious hardware constraints include Direct Access Storage Device (DASD) type and number, as well as numbers of channels. These constraints serve to bound the extent to which data allocation can be accomplished across different DASD for efficiency of access purposes. The most significant software constraints are those related to the DBMS used to implement the design. Examples of software constraints that can impact physical database design include page or block size range, maximum numbers of record occurrences in a file or data class, maximum numbers of record types, maximum number of relationships between record types of data classes. The types of metadata needed would include:

- catalog of hardware constraints
- catalog of software constraints

6.3.1.6 Expansion Factors. This area addresses the problem of arriving at a physical database design whose realization is capable of gracefully expanding to accommodate more data of the same or a different type. This impacts the decisions on file loading, partitioning, clustering, and access methods. Data needed to support decisions on expansion factors include:

- identification of data types, data occurrences and data mappings which are subject to expansion
- the order of magnitude of expansion of each of these items
- the time frame involved

6.3.1.7 Physical Record Size. This area addresses the problem of determining the size of the physical record which will be the database system's unit of storage and retrieval to/from secondary storage media. Limitations may be imposed by the database system of the secondary storage. Large physical records (blocks) may restrict the number concurrently available in main storage while reducing the need to transfer blocks. The physical record size design decision is based on these factors and:

- amount of data needed to satisfy a retrieval request
- probability of accessing more than one record in a block
- block loading density
- efficiency of the secondary storage system

6.3.1.8 Partitioning. Segmenting a record type into two or more subrecord types is record partitioning. Partitioning a file of records into two or more subfiles is file partitioning. Record and file partitioning may occur for performance or privacy/security reasons. Input to a partition decision process may include:

- Attribute and subfile reference frequencies
- Attribute length and subfile sites
- Attribute and subfile privacy/security constraints

6.3.1.9 Clustering. Intra file clustering involves the selection of a key on which records of a file are to be sequenced. Interfile clustering involves placing records of one file (e.g., member records) physically close or adjacent to records of another file (e.g., owner records). Input to the clustering decision process is:

- Operations on single files and the frequencies of these operations; operations on interfile linkages and the frequencies of these operations;
- List of keys that are candidates for intra file clustering.

6.3.1.10 Mapping

6.3.1.10.1 Definition of Mapping: A specification of how objects and operations at one level of abstraction correspond to objects and operations at another level of abstraction. Examples of levels are: user level (external schema), logical level (conceptual schema), and physical level (internal schema). Examples of objects are entities, relationships, attributes, pages, and pointers. Examples of operations are "find owner" and "follow pointer."

Top-down database design involves designing logical objects and then mapping them into physical objects based upon performance criteria, etc. The DDS should document the mapping.

6.3.1.10.2 Definition of Physical Data Independence: The ability to change the physical database design while preserving the original logical database design by changing the logical-physical mapping.

6.3.1.11 Recovery and Restart. This area concerns the physical database design issues relative to insuring that the database, either in whole or in part, can be recovered and restarted within the required time frame. Physical database level design activities that can be influenced by recovery/restart considerations include data partitioning and planned data redundancy. Recovery and restart requirements are established for data classes with respect to users. The basic question to be answered is, "How long can a user afford to be without data support?" The answer establishes the maximum time frame within which recovery and restart must be accomplished. Metadata needed to support physical design decisions related to recovery and restart include:

- maximum time that a data class can be unavailable
- identification of data classes that constitute minimum working sets of data classes, such that all data classes of the set must be available for effective processing to take place

6.3.2 Physical Allocation

6.3.2.1 Space Constraint. Information on the available storage space is necessary to support decisions on the allocation and distribution of the data class occurrences across the available DASD. This information is constantly changing as the available DASD capacity changes due to allocation and deallocation actions. Metadata needed here include:

- number of data classes (sets) that can be defined for each DASD unit
- available space for data storage for each DASD unit

6.3.2.2 Space Allocation. This involves the assignment of files to storage media and the allocation of storage space for these files. A goal of space allocation is to achieve an efficient and balanced utilization of available storage devices. Input to the space allocation process may include:

- size of each file
- growth rate of each file
- access frequencies of each file
- storage capacity of each device

171

6.3.3 Predictive Validation.

Analytic models, simulation models, and prototype databases are tools that are used as design aids. Each has its strengths and weaknesses. Predictive validation involves the use of these tools to confirm that the specific design constraints have been satisfied.

6.3.3.1 Analytic Models. Analytic models are efficient design tools. Equations that estimate performance indications are defined and can, in most circumstances, be evaluated efficiently. Analytic models have been developed to simplify decisions concerning partitioning, clustering, physical record size, and concurrency, among others. The greatest limitation of analytic models is that many problems are too complex to be addressed using analytic techniques. In such cases, simulation and prototyping may be used. Input to analytic models may include:

- Logical data structure and its descriptive characteristics (e.g., number of entities of a particular type)
- Transactions that are to be processed and transaction frequency

Simulation is a powerful tool for validating physical designs. It can be used to access global feasibility (i.e., will the implemented design meet the processing and storage requirements). Typically, simulation is used to predict the behavior of complex systems. Simulation experiments are designed to demonstrate the performance of database systems under various conditions of processing load, logical and physical data organizations and database size. Data needed to support the simulation of a physical design include:

- the logical data structure
- the physical data organization
- transactions to be processed
- the DBMS software
- placement of data on secondary storage

6.3.3.2 Prototype Approach. This area deals with the design and specification of a prototype based method for validating that a physical database design satisfies the established requirements and observes the stipulated constraints. A prototype based validation process is one that includes database load, database maintenance and database use (basically applications software) software as well as a representative database. The objective of the use of the software with respect to the prototype database is to demonstrate that all required data types and relationships are created and useable for generating the needed information.

The prototype database does not have to be equivalent to the expected live data with respect to quantity or volume of data. It does have to be logically equivalent to the live database. This means that each unique data value and combination of logical conditions should be represented. The metadata needed to support this area include:

- logical data structures defining logical entities and relationships
- entity keys used for identification reference and cross reference
- response requirements by data class and operation
- report requirements
- transactions that must be processable
- access control, privacy, and security requirements

6.4 IMPLEMENTATION

This is the phase where the user loads the database that has been logically and physically designed--in other words, the "moment of truth."

6.4.1 Conversion Plan

Whether converting from manual records, standard files, or another DBMS, it is essential to have worked out a conversion plan for the data in advance (see Data Base Directions: The Conversion Problem, NBS Special Publication 500-64.)

6.4.2 Data Dictionary System Assistance

Whether an active or passive DDS is in use, the DDS should reflect information about the new database design, and possibly, the older design. Right now, the DDS can provide a road map for implementation. In the future, as Ronald G. Ross asks in Data Dictionaries and Data Administration: Concepts and Practices for Data Resource Management (AMACOM, 1981; excerpts printed in Computerworld, Vol. XIV, No. 38, September 17, 1980, pp. 36-47), why shouldn't it be possible to use an active DDS system directly for assistance in the automatic installation of the database.

6.4.3 Prototyping

Just as the physical design of the database can be tested and modified during the design phase, so can the initial implementation itself be a prototype. There are many examples of interactive decision support systems and "user friendly" database management systems, usually for smaller-scale databases, where efficient performance is not as important a consideration as fast implementation. In these instances, the prototype may turn out to be a satisfactory final solution.

6.4.4 Complexity

As the richness of a data structure increases or the sheer physical size of a very large database leads to a complex physical arrangement, something more than prototyping is necessary. For example, a test database, representing a partial load of the database, might be established and tested before a complete database is loaded.

6.4.5 Validation of the Design

Satisfactory performance of a prototype, a test database or a complete database in a production system environment is necessary to validate the physical design of the database. Despite the claims of vendors, the hopes of in-house technicians and the results of today's analytical or simulation models, it is not until final implementation that users know they have succeeded. Unfortunately, except for quickly installed prototypes, it is usually very costly and time-consuming to get to the implementation phase. It can be disastrous to get this far only to find that one must go through a major redesign to obtain adequate response times or avoid installing 20 additional disk drives beyond the 10 estimated! Often, by this time there are waiting application development teams and users who will be extremely unhappy at further delay.

6.4.6 Data Validation

The process of loading a database may turn up significant discrepancies from information gathered during the Requirements phase. It can be most embarrassing to discover at the last minute that there are, in fact, twice as many vendors as planned or duplicate names or codes or simply that the old data is "dirty" or full of obsolete codes. Again, good systems anaysis work during the Requirement phase and Physical Design phase can help minimize discrepancies. Additionally, in today's data processing environments, users may find it worthwhile to make statistical or editing runs against the data prior to conversion (or even prior to Physical Design), using tools such as general purpose file management utilities. Since some of today's DDS's already provide for limited statistics and data item editing criteria, it would seem possible in the future to establish and use such information in a better organized, more automatic way with the DDSs during the Requirements and Physical Design phases. Also, if the future state of the art someday allows us to achieve automatic loading of the database using the DDS, we might reasonably expect "editing concurrent with loading."

6.4.7 Stress Testing

When a new database has been loaded, but before it has been fully turned over to operations, it may be advisable to conduct a stress test—especially where on-line response times are critical or for a complex or very large database. For this, the designer/implementor should pre-plan with end-users and operations a high volume test of transactions or concurrent terminals' access of the database. The idea is to flush out inherent performance problems and take corrective action before going "live."

6.4.8 Confirmation

While the data and physical design can be validated in ways suggested above, final confirmation of successful design and performance cannot be made until the database has been operational in a production environment for an appropriate length of time. Performance measures should be pre-established with users' and operations' prior agreement so that later on, in the Operations phase, the monitoring activity will have something to compare against. Establish the measures before going "live" to avoid the problem of a "moving target" as users become more familiar with performance characteristics and capabilities. In the future, it would be desirable to have a new generation DDS with the capability of carrying both expected performance measures and monitored performance results, so that exception reports could be directed to the DBA for investigation and corrective action!

6.5 OPERATIONS

This phase of the database lifecycle is the period of operational use in meeting the needs of the enterprise. If the database is successful, this will be by far the longest phase. During this phase the physical characteristics of the database and the way it is used (i.e., the user's view of virtual records) will continuously and gradually change. This implies the need to continuously monitor performance and its physical attributes, to systematically collect and organize these data, and to selectively adjust the physical database design. Note that such changes in physical characteristics are not uniform across either logical or physical subsets of the database. The panel recognized the DDS as the logical mechanism for organizing these measurements of physical attributes.

6.5.1 Gathering Performance Data

- Who invokes the gathering? (DBA, automatic)
- When invoke the gathering? (always, sometimes)
- What gather, and how?
- Compare actual characteristics of data and usage with Phase I estimates.
- Granularity of measurement.
- What is overhead of gathering?

6.5.2 Aggregating Performance Data

- How?
- Time granularity.

6.5.3 Reporting Performance Data

- Who invokes the reporting? (DBA, automatic)
- How often report?
- How present? (machine-readable and human-readable)
- Who has access to data?

6.5.4 Analyzing Performance Data

- What measures of performance?
- Compare with desired/predicted performance.
- Database performance vs. whole system performance.
- How determine what caused a given aspect of performance?

6.5.5 Performing Maintenance

- Who invokes the maintenance? (DBA, automatic)
- Invoke under what circumstances? (periodic (e.g., ISAM), upon demand (e.g., VSAM))
- How perform? (incremental (e.g., VSAM), offline unload/reload or copy (e.g., ISAM), offline in-place (e.g., an IDMS utility), concurrently with usage (research topic))

Note: Most of this applies also to one-shot reorganization (definitional changes in evolution phase).

6.5.6 DDS Usage.

The DDS is the vehicle for documenting the answers to some of these questions:

Document: Desired performance from Phase I
Predicted performance from Phase II
Actual performance from this phase
Estimated characteristics of data and usage from Phase 1
Actual characteristics of data and usage from this phase.

6.6 EVOLUTION

Virtually any requirement in Phase I can change. This may require logical redesign (restructuring) or it may require just physical redesign (reorganization).

Poor performance may require physical redesign. We then continue through the cycle of re-implementation (perhaps) and new operation. This discussion focuses upon physical re-design.

Different types of evolution, in different circumstances, have different degrees of difficulty. Physical data independence eases the problems of reorganization, since users and applications are unaffected (except via performance changes). Without physical data independence, reorganization to improve performance can affect applications, thus requiring potentially costly modification. This modification can range from recompilation to reprogramming.

As an example of evolution, suppose that in an energy shortage, a company decides that it must access personnel records by home zip code for the purpose of forming car pools. To accomplish this efficiently, the DBA decides to create a new secondary index. Depending upon the DBMS, the difficulty of doing this ranges from (1) adding a new table to (2) recompiling existing tables and re-loading the database.

As another example, adding a new application or changing the frequency of execution of an existing application may result in a new performance bottleneck, which then requires reorganization.

6.6.1 Definition of Two Types of Reorganization

6.6.1.1 One Shot Reorganization. Change definition of data; not performed repeatedly (at least not for a database whose characteristics are stable). Examples: Change from indexing to hashing; move to a new device.

6.6.1.2 Maintenance. Change just occurrences, not definitions. Performed repeatedly (periodically or upon demand). Examples: eliminate overflow in ISAM; VSAM split; compact to recover space.

6.7 END GAME

At some point in the life of a database, operations will eventually be shifted (partially or in total) to a distinctly different mechanism (a file or another database). This is a conversion similar to the originating conversion and as such it is necessary to establish correspondence between the old and the new data forms.

An analogous conversion process will normally be initiated at the outset of the operational phase; specifically, data archival. The distinction between these two types of conversion is subtle - totality versus selectivity. In those rare instances when data is no longer captured the end-game archiving of data is, in fact, a total conversion.

Items for consideration relative to the termination of a system:

- Graceful exit
- Retention of Records (internal, external)
- Retention of record format
- Accessibility of records
- Retention of access programs
- Impact on other systems
- Changes to structure to support "new" retrieval frequencies

Entities/Attributes related to the demise of a system:

- Users
- Retention time for data
- System requirements (operating system, DBMS, etc.)
- Programs to access data
- Data
- Data aggregates (Groups)
- Structure

7. PARTICIPANTS

The following is a list of panel members and other workshop participants.

Prof. Donald Batory Logical Design Physical Design	Computer Systems Research Group University of Toronto 121 St. Joseph Street Toronto, Ontario, CANADA M5S 1A1
Mr. Thomas Baumgartner Physical Design	DBD Systems 1500 North Beauregard Alexandria, VA 22311
Mr. John Berg Planning Committee	Mgr., Computer and Info. Technology Standard Oil (Indiana) 200 E. Randolph Chicago, IL 60601
Ms. Susan M. Bidwell Standards and Controls	Blue Cross Blue Shield 233 North Michigan Avenue Chicago, IL 60601
Prof. Michael L. Brodie Logical Design	Computer Science Department University of Maryland College Park, MD 20740
Prof. Joseph Brownsmith Physical Design	512 Weil Hall University of Florida Gainesville, FL 32611
Mr. Milt Bryce Planning Committee	M. Bryce & Associates, Inc. 1248 Springfield Pike Cincinnati, OH 45215
Prof. Peter Buneman Logical Design	Moore School of Electrical Eng. University of Pennsylvania Philadelphia, PA 19104

Mr. James H. Burrows Planning Committee	Director, Institute for Computer Sciences and Technology National Bureau of Standards ADMIN A200 Washington, DC 20234
Mr. Richard G. Canning Planning Committee	Canning Publications, Inc. 925 Anza Avenue Vista, CA 92083
Mr. Carl R. Clark Uses of the DDS	Texas Education Agency Management Information Center 210 E. 11th Austin, TX 78701
Prof. Eric Clemons Logical Design	Department of Decision Sciences The Wharton School University of Pennsylvania Philadelphia, PA 19104
Dr. Craig Cook Standards and Controls	Arthur Young & Co. 1025 Connecticut Ave., N.W. Washington, DC 20036
Mr. Thomas J. Cook Logical Design	Tektronix P.O. Box 500 Mail Stop 50-384 Beaverton, OR 97077
Mr. Gary D. Craig Logical Design	IBM Dept. 997/B622-5 P.O. Box 12195 Research Triangle Park, NC 27709
Mr. Robert M. Curtice Logical Design	Arthur D. Little Acorn Park Cambridge, MA 02140
Prof. Al Dale Uses of the DDS	Dept. of Computer Science University of Texas at Austin Austin, TX 78712

Mr. Charles Drumm Uses of the DDS	Reliance Insurance Co. 4 Penn Center Plaza 16th Floor Philadelphia, PA 19103
Mr. Paul L. Fehder Logical Design	M49/B35 IBM Santa Teresa Laboratory 555 Bailey Avenue San Jose, CA 95150
Dr. Dennis Fife Planning Committee	Systems and Applied Sciences Corp. 6811 Kenilworth Avenue Riverdale, MD 20737
Mr. Thomas J. Fitzgerald Standards and Controls	Merrill Lynch & Co., Inc. One Liberty Plaza 165 Broadway New York, NY 10080
Mr. Donald Fox Uses of the DDS	GE Information Serv. Co. (4NW) 401 N. Washington St. Rockville, MD 20850
Mr. James Fry Logical Design Physical Design	Graduate School of Business Admin. Data Systems Research Group University of Michigan Ann Arbor, MI 48109
Mr. George Gajnak Standards and Controls	Cincom Systems 2300 Montana Avenue Cincinnati, OH 45211
Ms. Gayle S. Gillingham Uses of the DDS	U.S. Geological Survey Mail Stop 115 Reston, VA 22092
Mr. Richard H. Godlove Logical Design	Monsanto Chemicals Company Management Inf. Systems Dept. 800 North Lindbergh Blvd. St. Louis, MO 63166

Dr. Alan Goldfine Proceedings Editor	Institute for Computer Sciences and Technology National Bureau of Standards TECH A265 Washington, DC 20234
Mr. John Gosden Keynote Speaker	VP-Telecommunications Equitable Life Assurance Co. 1285 Avenue of the Americas, 9D New York, NY 10019
Mr. Robert R. Hegland Uses of the DDS	NARDAC WASH DC - Code 3024 Washington Navy Yard, Bldg. 196 Washington, DC 20374
Ms. Margaret A. Herrick Physical Design	Margann Associates 205 Hamilton Street Cambridge, MA 02139
Mr. Thomas R. Hester Logical Design	Intel Systems Corp. 12675 Research Blvd. Austin, TX 78766
Mr. Scott Hightower Standards and Controls	Petro Lewis Corporation Petro-Lewis Tower 717 17th Street Denver, CO 80201
Dr. Barron C. Housel Logical Design	IBM Corporation Department E-96 P.O. Box 12195 Research Triangle Park, NC 27709
Dr. David K. Jefferson Logical Design	Naval Ship Research and Dev. Center Code 1821 Bethesda, MD 20084
Mr. Seymour Jeffery Planning Committee	TRW Defense and Space Systems Group Room W1/6554 7600 Colshire Drive McLean, VA 22102

Prof. Beverly Kahn Logical Design	Boston University School of Management 212 Bay State Road Boston, MA 02215
Dr. Larry Kerschberg Logical Design	Bell Laboratories WB 1G-307 Holmdel, NJ 07733
Mr. Allan J. LaRue Uses of the DDS	Arizona Public Service Co. P.O. Box 21666 - Station 1285 Phoenix, AZ 85036
Mr. Thomas Lee Uses of the DDS	4500 S. Four Mile Run Drive Apt. 514 Arlington, VA 22204
Dr. Henry C. Lefkovits Standards and Controls	Alpha Omega Group, Inc. P.O. Drawer M Harvard, MA 01451
Ms. Belkis Leong-Hong Standards and Controls	OASD (C) 17 Cullinan Court Gaithersburg, MD 20878
Lt. Col. Marv D. Lerfald Uses of the DDS	5117 Leesburg Ct. Woodbridge, VA 22193
Mr. John K. Lyon Physical Design	General Electric Co. Bldg 23EW 1285 Boston Avenue Bridgeport, CT 06602
Mr. Daniel B. Magraw Uses of the DDS Planning Committee	Department of Administration State of Minnesota 208 State Administration Building St. Paul, MN 55155
Mr. John Mayson Uses of the DDS	GTE Services Corp. P.O. Box 1548 Tampa, FL 33601

Mr. Donald J. McCaffrey Uses of the DDS	PRC Govt. Info. Systems 7432 Long Pine Drive Springfield, VA 22151
Prof. Dennis McLeod Logical Design	Salvatori 206 Computer Science Department University of Southern California Los Angeles, CA 90007
Mr. Mike Meyer Standards and Controls	Honeywell Information Systems 7900 Westpark Drive MS702 McLean, VA 22102
Mr. John Midgley Physical Design	Honeywell m/s C73 P.O. Box 6000 Phoenix, AZ 85005
Prof. Jack Minker Planning Committee	Department of Computer Science University of Maryland College Park, MD 20742
Mr. Richard N. Mixon Standards and Controls	Middle South Services P.O. Box 249 Gretna, LA 70053
Ms. Joan Monia Physical Design	Digital Equipment Corp. 191 Spring Street Lexington, MA 02173
Mr. Thomas E. Murray Physical Design	Mgr., Infor. Services Development Del Monte Corporation P.O. Box 3575 San Francisco, CA 94119
Prof. Shamkant Navathe Logical Design	512 Weil Hall Dept. of Computer and Information Sciences University of Florida Gainesville, FL 32611

Mr. Has Patel Uses of the DDS	MSP Inc. 21 Worthen Road Lexington, MA 02173
Mr. William E. Perry Standards and Controls	9222 Bay Point Drive Orlando, FL 32811
Mr. Mayford Roark Planning Committee	Ford Motor Company The American Road Dearborn, MI 48121
Mr. Bruce W. Rollier Logical Design	23 Whitman Pl. Hillsdale, NJ 07642
Ms. Susan L. Rosenbaum Uses of the DDS	AT&T 50-52 C278 P.O. Box 3509 New Brunswick, NJ 08903
Ms. Zella Ruthberg Standards and Controls	Institute for Computer Sciences and Technology National Bureau of Standards TECH B266 Washington, DC 20234
Mr. Roy G. Saltman Uses of the DDS	Institute for Computer Sciences and Technology National Bureau of Standards TECH A265 Washington, DC 20234
Mr. Daniel B. Schneider Uses of the DDS	U.S. Department of Justice 5843 Potomac Avenue, N.W. Washington, DC 20016
Mr. Richard D. Secrist Logical Design	Systems Dev. Support - Exploration Standard Oil Company (Indiana) P.O. Box 591 Tulsa, OK 74102

Mr. Dennis Shaw
Standards and Controls
U.S. General Accounting Office
FGMSD-ADP
Room 6011
441 G Street N.W.
Washington, DC 20548

Dr. Gary Sockut
Physical Design
Institute for Computer Sciences
and Technology
National Bureau of Standards
TECH A265
Washington, DC 20234

Mr. Jim Swager
Uses of the DDS
Honeywell Information Systems
7900 Westpark Drive
McLean, VA 22102

Ms. Ruth Thorburn
Physical Design
United Airlines
M/C EXOKA
P.O. Box 66100
Chicago, IL 60666

Mr. G. David Vincent
Standards and Controls
Alpha Omega Group, Inc.
World Building
8121 Georgia Avenue
Silver Spring, MD 20910

Mr. Ronald F. Voell
Standards and Controls
Foremost-McKesson, Inc.
One Post Street
San Francisco, CA 94104

Prof. Jay Louise Weldon
Logical Design
48 Tuxedo Road
Montclair, NJ 07042

Prof. Gio Wiederhold
Logical Design
Computer Science Dept.
Stanford University
Margaret Jacks Hall #38
Stanford, CA 94305

Dr. Jerry Winkler
Uses of the DDS
5424 Southport Lane
Fairfax, VA 22032